NASA-CR-190895

SBIR-02.07-5630
Release date 7/20/92

AN ASSESSMENT OF LASER VELOCIMETRY IN HYPERSONIC FLOW

145752

p. 227

FINAL REPORT
NAS 2-12853

(NASA-CR-190895) AN ASSESSMENT OF LASER VELOCIMETRY IN HYPERSONIC FLOW Final Report (Complere) 227 p

N93-21556

Unclas

G3/02 0145752

COMPLERE INC.
P.O. BOX 1697
PALO ALTO, CA

SEPTEMBER 1992

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SBIR RIGHTS NOTICE

1.0 ABSTRACT

Although extensive progress has been made in computational fluid mechanics, reliable flight vehicle designs and modifications still cannot be made without recourse to extensive wind tunnel testing. Future progress in the computation of hypersonic flow fields is restricted by the need for a reliable mean flow and turbulence modeling data base which could be used to aid in the development of improved empirical models for use in numerical codes. Currently, there are few compressible flow measurements which could be used for this purpose. In this report, the results of experiments designed to assess the potential for laser velocimeter measurements of mean flow and turbulent fluctuations in hypersonic flow fields are presented. Details of a new laser velocimeter system which was designed and built for this test program are described.

2.0 INTRODUCTION

Current hypersonic flow field instrumentation is insufficient to meet present and future ground test requirements. Measurements are required to establish the basic physical mechanisms and turbulence models required for reliable prediction of transitional and turbulent hypersonic flow fields.

In recent years, experimental methods in lower speed regimes have also made significant advances due primarily to the availability of high power lasers. Their introduction has enabled the field of laser velocimetry to expand from low speed, small scale, closely controlled laboratory applications to the measurement of compressible flows in large scale wind tunnels (Ref. 1). The advent of the laser velocimeter allows us to measure velocity fluctuations directly in a linear, non-intrusive manner. Of particular value is the capability it offers to measure some of the compressible turbulent shear stresses, since this is an impractical task with hot wires (Ref. 2).

However, before laser velocimetry can be extended to hypersonic flow,

some basic questions must be addressed. The primary question is that of particle size requirement for reliable response combined with adequate Mie scattering. Practical assessments must therefore be made of flow seeding capability and the potential for laser velocimetry in hypersonic flows.

3.0 EXPERIMENTAL DETAILS

The laser velocimeter investigation was conducted in the NASA Ames 3.5 Ft. Hypersonic Wind Tunnel. In this facility, high-pressure air flows through a pebble bed heater and then through an open jet test section to lower pressure spheres. The tests were conducted at a nominal freestream Mach number of 7 and a freestream Reynolds number of 3 million per foot. The test model used in this study was a 10° cone-ogive-cylinder which was 79 inches long and 8 inches in diameter (Ref. 3). Measurements were made in the local freestream above the model and in the zero pressure gradient boundary layer flow on the cylindrical portion of the model. Measurements were also made across an oblique shock wave generated by the introduction of a 20 deg. flare installed 55 inches from the nose. A seed particle generator and injectors were designed and installed in the facility. The particles were injected through a thermocouple port into the plenum just upstream of the throat. A schematic of the seeding system and seeder operational procedures are given in Figure 1 and Table 1 respectively. Seed particles and seed mixtures detailed in Ref. 4 were used during the tests. The two component, forward scatter, fringe mode laser velocimeter system, which was used for the flow field measurements, utilized the 4880 and 5145 Angstrom lines of an argon-ion laser. Details of the optical system are presented in Table 2. Details of the traverse control and data acquisition systems are described in Ref. 5.

4.0 TEST RESULTS

Initially, measurements were confined to the local freestream until seeder mass flow rates and procedures were optimized for data rate and signal to noise ratio. Figure 2 shows examples of signal quality in *wind off* and *wind on* situations. Clearly, signal quality, visibility and fringe crossings were adequate in the freestream hypersonic flow. On occasion, data rates of more than 100,000 per second were measured. Figure 3 shows the mean boundary layer flow results along with the mean profile measurements which were obtained from previous conventional probe measurements (Ref. 6). Although, as expected, the signal to noise ratio decreased close to the wall, the good agreement between the two measurement methods confirms the seed particle response for mean velocity measurements in the zero pressure gradient boundary layer.

The results of a more stringent test of the particle response and the laser velocimeter measurements are shown in Figure 4 where the zero pressure gradient axial and vertical turbulence measurements are presented. These data show similarities in levels and trends to previous incompressible test results. The streamwise turbulence component has a pronounced maximum close to the wall whereas the vertical component, which is approximately half the axial value, is relatively flat in the wall region. These similarities are not altogether surprising since previous hot wire turbulence convection velocity measurements (Ref. 3) showed that the relative velocity between the disturbances and the local mean flow was always subsonic which allows the turbulent bursts to propagate as they would in an incompressible flow.

The axial component measurements are also compared with Klebanoff's incompressible results and previous hot wire hypersonic measurements in Figure 5. There is reasonably good agreement between the hypersonic laser velocimeter and incompressible hot wire data when normalized by the wall friction velocity. This is in contrast to previous hot wire compressible flow results, reviewed in Ref. 6, which show a monotonic decrease with increasing Mach number. However, all these past hot wire results have been evaluated

assuming zero pressure fluctuations which we would expect to become more important with increasing Mach number (Ref. 2). The turbulent velocity cross correlations are presented in Figure 6, which shows the variation of the turbulent velocity correlation coefficient across the boundary layer. The maximum value of approximately -0.4 is in close agreement with incompressible shear layer observations.

The most stringent test of particle response was made by perturbing the flow and measuring the particle velocity variation across an oblique shock wave and shear layer generated by the introduction of a 20 deg. flare. Unfortunately, these attempts to determine particle response were complicated by the proximity of the shock to the shear layer on the flare and by shock boundary/layer interaction instabilities. The results of a scan taken 2 inches above the model surface are presented in Figure 7 which shows the measured mean streamwise velocity and flow angularity distributions through the shock and shear layer region compared with conical flow theory and shadowgraph measurements of the shock location. The location of the measured mean velocity gradient is in good agreement with the shadowgraph shock location and the velocity change across the shock is comparable to conical flow predictions until the shear layer is encountered. The flow angularity measurements are consistent with conical flow predictions and the experimental flare angle. These comparisons indicate adequate particle response since some of the velocity and flow angularity gradient discrepancies across the shock are probably caused by small scale, time dependent oscillations of the shock wave about its mean location. Indeed, attempts to measure particle response across the 30 deg. shock wave were unsuccessful as the increased tunnel blockage led to excessive flow field instabilities and extensive shock motions.

The velocity probability density distributions, shown in Figure 8, are narrow in the freestream ahead of the shock where the turbulence level is low and wider in the more turbulent region within the shock layer. They are clearly bimodal in the region of the time averaged shock location. These bimodal distributions are of most interest as they give a clear indication of particle

response in hypersonic flow. The bimodal distributions shown in Figure 8 are due to shock wave fluctuations around its mean location. Thus, if the particles follow the flow, the two, bimodal peaks should be a measure of the velocity change across the shock. Since, when the instantaneous shock location is upstream of the focal volume, particles will register the lower velocity behind the shock and, when the focal volume is upstream of the instantaneous shock location, the higher freestream velocity will be recorded.

The shift from the dominant freestream peak ahead of the shock as the probe volume is traversed towards the model, is a measure of the probability of shock passage through the focal volume. The location of the most symmetrical bimodal distribution is the most likely, time-average shock location. Thus, from these velocity probability density distributions we can determine the particle velocity change across the shock and estimate the mean shock location above the plate. These results compare well with theoretical velocity change predictions and optical observations of the mean shock location.

These measurements can also be used to assess seed particle response and dynamics in hypersonic flow. Using the measured velocity change and calculated transit time through the shock wave region, seed particle response characteristics can be calculated. These calculations show that the measured particle response is equivalent to that of a 0.3 micron, specific gravity 1.0 sphere which undergoes a deceleration of almost seven million times the acceleration due to gravity; ie. 7 Mg. This size and acceleration is consistent with hypersonic modifications to the Stoke's drag law. Since, in hypersonic flow the particle drag coefficient is inversely proportional to the particle Reynolds and Knudsen numbers.

5.0 <u>CONCLUDING REMARKS</u>

Diagnostic tools are available to attempt the measurement of turbulent hypersonic flows, an area where comprehensive studies are lacking. Comparisons of new laser velocimeter turbulence measurements with previous hot wire results indicates that past data reduction assumptions can result in

significant measurement errors in hypersonic flows. It is felt that these new test results are the most convincing evidence to date of particle response in hypersonic flow. They clearly show that attempts to assess seed particle response must involve detailed studies of the velocity probability distributions. Particle response assessments inferred from conventional time-averaged velocity measurements could well be flawed by their failure to account for the hidden, adverse effects of large-scale, time-dependent mean flow variations which, on closer examination, may well manifest themselves in the velocity probability density distributions. Clearly, extensive work is still needed to establish a reliable data base for turbulence modeling and to define the reliable ranges of laser anemometer application.

6.0 REFERENCES

1. Owen, F. K., *Application of Laser Velocimetry to Unsteady Flows in Large Scale, High Speed Wind Tunnels*, ICIASF '83 Record, IEEE Publication 83CH1954-7, 1983.

2. Owen, F. K. and Fiore, A. W., *Turbulent Boundary Layer Measurement Techniques*, AFWAL-TR-86-3031, 1986.

3. Owen, F. K. and Horstman, C. C., *On the Structure of Hypersonic Turbulent Boundary Layers*, J. Fluid Mech., Vol. 53, pt. 4, p. 611, 1972.

4. Seegmiller, H. L., *Seeding Subsonic, Transonic and Supersonic Flows with 0.5 Micron Polystyrene Spheres*, NASA CP 2393, 1985.

5. Complere Inc., *The NASA Ames 3.5-Ft. Hypersonic Wind Tunnel Laser Velocimeter System*, Contract Report 92-0401, April 1992.

6. Owen, F. K., Horstman, C. C., and M. I. Kussoy, *Mean and Fluctuating Flow Measurements of a Fully Developed, Non-adiabatic Hypersonic Boundary Layer*, J. Fluid Mech., Vol. 70, pt. 4, p. 393, 1975.

Table 1. Seeder Operation.

LDV SEEDER OPERATION : LIQUID SEED

All manual and solenoid valves closed except #10 (drain). Liquid seed nozzle installed in heater port. Flex hose lines attached to liquid seed tank. Control valves backed off.

FILL TANK

1) Open MV-5 (manual).
2) Open MV-4 (manual).
3) Fill tank with liquid seed until it flows out of drain valve.
4) Close MV-5, MV-4, and drain #10.

PRESSURIZE LIQUID SEED TANK AND SEED LINE TO HEATER

5) Open valve to 3000 PSIA nitrogen bank.
6) Open MV-3 (manual) and CV-3 (solenoid).
7) Adjust FCV-3 (control valve) until pressure is above tunnel total pressure. Use pressure gage PT-3 to adjust. N.B. FCV-3 valve will be incrementally adjusted to achieve optimum seeding during run.
8) Open MV-2 (manual).
9) Open CV-2 (solenoid).

PRESSURIZE 3000 PSIA AIR LINE TO HEATER

10) Open valve to 3000 PSIA air line. Open CV-1 (solenoid).
11) Adjust FRV-1 and FCV-1 (control) so pressure read by gage PT-1 is 100 PSI above tunnel total pressure.
12) Close CV-1 (solenoid) and open MV-1 (manual).

START BLOWDOWN

IN CONTROL ROOM

13) When tunnel conditions are met, open CV-1 (solenoid).
14) Adjust control valves FCV-3 and FCV-1 so that seed pressure and air pressure are higher than tunnel operating pressure. Use LDV Data Acquisition System plus oscilloscope to determine optimum seeding.

END BLOWDOWN

IN CONTROL ROOM

15) Back off FCV-3 (control). Close CV-3 (solenoid).
16) Back off FCV-1 and FRV-1 (control). Close CV-1 (solenoid).

BY HEATER

17) Close MV-1 and MV-3 (manual).
18) Close valves to 3000 nitrogen and air lines.
19) Open MV-5 (manual) to relieve pressure in seeder tank to heater flue.
20) When flow stops, close MV-5, open drain #10, remove and/or replace seeder nozzle and seed filter (#6).

Table 1. Seeder Operation Continued.

LDV SEEDER OPERATION : DRY SEED

All manual and solenoid valves closed except #10 (drain). Flex hose lines attached to dry seed tank. Control valves backed off. Liquid seed filter screen removed to prevent clogging.

FILL TANK

1) Unscrew dry seed filler cap from dry seed tank.

2) Fill with dry seed, and replace filler cap.

3) Hook up to flex hose lines.

4) Close drain valve (#10).

5) Follow steps 5 to 18 of liquid seed operation then complete the following steps 6 & 7.

6) Open MV-6 (manual) to relieve pressure in seeder tank to heater flue.

7) When flow stops, close MV-6, open drain #10, remove and/or replace seeder nozzle.

Table 2. Optical Details.

Parameter	Symbol or Equation	Value	Units
Wavelength	Lambda	5145	Å
Focal Length (transmitting lens)	Ft	0.7620	meters
Focal Length (receiving lens)	Fr	0.7620	meters
Focal Length (lens to fiber)	Ff	0.7620	meters
Aperture Diameter at Fiber	Df	0.0006	meters
Receiving Side Lens Diameter	Ld	0.1524	meters
Beam to Receiving Lens Gap	Gap	0.0889	meters
Beam Separation at Transmitting Lens	Bt	0.007 938	meters
Beam Diameter at Lens	Dl	0.002 200	meters
Convergence Full Angle	Tf = 2*ATAN(Bt/2/Ft)	0.597	degrees
Convergence Half Angle	Th = 1*ATAN(Bt/2/Ft)	0.298	degrees
Fringe Spacing	X = Lambda/(2*SIN(Th)	0.000 049	meters
Number of Fringes in Probe Volume	Npv= Dpv/X	5	---
Off Axis Collecting Angle	Tc = Th+ATAN(Gap/Fr)+ATAN(Ld/2/Fr)	12.7	degrees
Beam Diameter at Waist	Dw = 4*Lambda*Ft/Pi/Dl)	0.000 227	meters
Beam Diameter at Probe Volume	Dpv= Dw/COS(Th)	0.000 227	meters
Length of Probe Volume	Lpv= Dw/SIN(Th)	0.043 565	meters
Probe Volume Effective Length	Vl = Df*Fr/Ff/SIN(Tc)	0.002 737	meters
Macrodyne Frequency	Fmac=Fringes*Clock/(Bin*2^(Range-0))	---	Hz
Bragg Frequency	Fbrag	40 000 000	Hz
Mixing Frequency	Fmix	0	Hz
Sign of Macrodyne Frequency	Smac	-1	---
Sign of Bragg Frequency	Sbrag	1	---
Sign of Mixing Frequency	Smix	1	---
Counter Clock Rate	Clock	1 000 000 000	Hz
Fringes Counted	Fringes	8	---
Total Frequency	Ftotal=Smac*Fmac+Sbrag*Fbrag+Smix*Fmix	---	Hz
Velocity	Velocity=X*Ftotal	---	m/s
Velocity Resolution	Resolution	---	m/s
Time in Focal Volume	T = ABS(Dpv/Velocity)	---	s
Number of Fringes Seen	Ns = Fmac*T	---	---
Power at fiber exit (nominal)	Power	0.3	Watts

Table 2. Optical Details Continued.

Parameter	Symbol or Equation	Value	Units
Wavelength	Lambda	4880	Å
Focal Length (transmitting lens)	Ft	0.7620	meters
Focal Length (receiving lens)	Fr	0.7620	meters
Focal Length (lens to fiber)	Ff	0.7620	meters
Aperture Diameter at Fiber	Df	0.0006	meters
Receiving Side Lens Diameter	Ld	0.1524	meters
Beam to Receiving Lens Gap	Gap	0.0889	meters
Beam Separation at Transmitting Lens	Bt	0.007 938	meters
Beam Diameter at Lens	Dl	0.002 200	meters
Convergence Full Angle	Tf = 2*ATAN(Bt/2/Ft)	0.597	degrees
Convergence Half Angle	Th = 1*ATAN(Bt/2/Ft)	0.298	degrees
Fringe Spacing	X = Lambda/(2*SIN(Th)	0.000 047	meters
Number of Fringes in Probe Volume	Npv= Dpv/X	5	---
Off Axis Collecting Angle	Tc = Th+ATAN(Gap/Fr)+ATAN(Ld/2/Fr)	12.7	degrees
Beam Diameter at Waist	Dw = 4*Lambda*Ft/Pi/Dl)	0.000 215	meters
Beam Diameter at Probe Volume	Dpv= Dw/COS(Th)	0.000 215	meters
Length of Probe Volume	Lpv= Dw/SIN(Th)	0.041 321	meters
Probe Volume Effective Length	Vl = Df*Fr/Ff/SIN(Tc)	0.002 737	meters
Macrodyne Frequency	Fmac=Fringes*Clock/(Bin*2^(Range-0))	---	Hz
Bragg Frequency	Fbrag	40 000 000	Hz
Mixing Frequency	Fmix	0	Hz
Sign of Macrodyne Frequency	Smac	-1	---
Sign of Bragg Frequency	Sbrag	1	---
Sign of Mixing Frequency	Smix	1	---
Counter Clock Rate	Clock	1 000 000 000	Hz
Fringes Counted	Fringes	8	---
Total Frequency	Ftotal=Smac*Fmac+Sbrag*Fbrag+Smix*Fmix	---	Hz
Velocity	Velocity=X*Ftotal	---	m/s
Velocity Resolution	Resolution	---	m/s
Time in Focal Volume	T = ABS(Dpv/Velocity)	---	s
Number of Fringes Seen	Ns = Fmac*T	---	---
Power at fiber exit (nominal)	Power	0.3	Watts

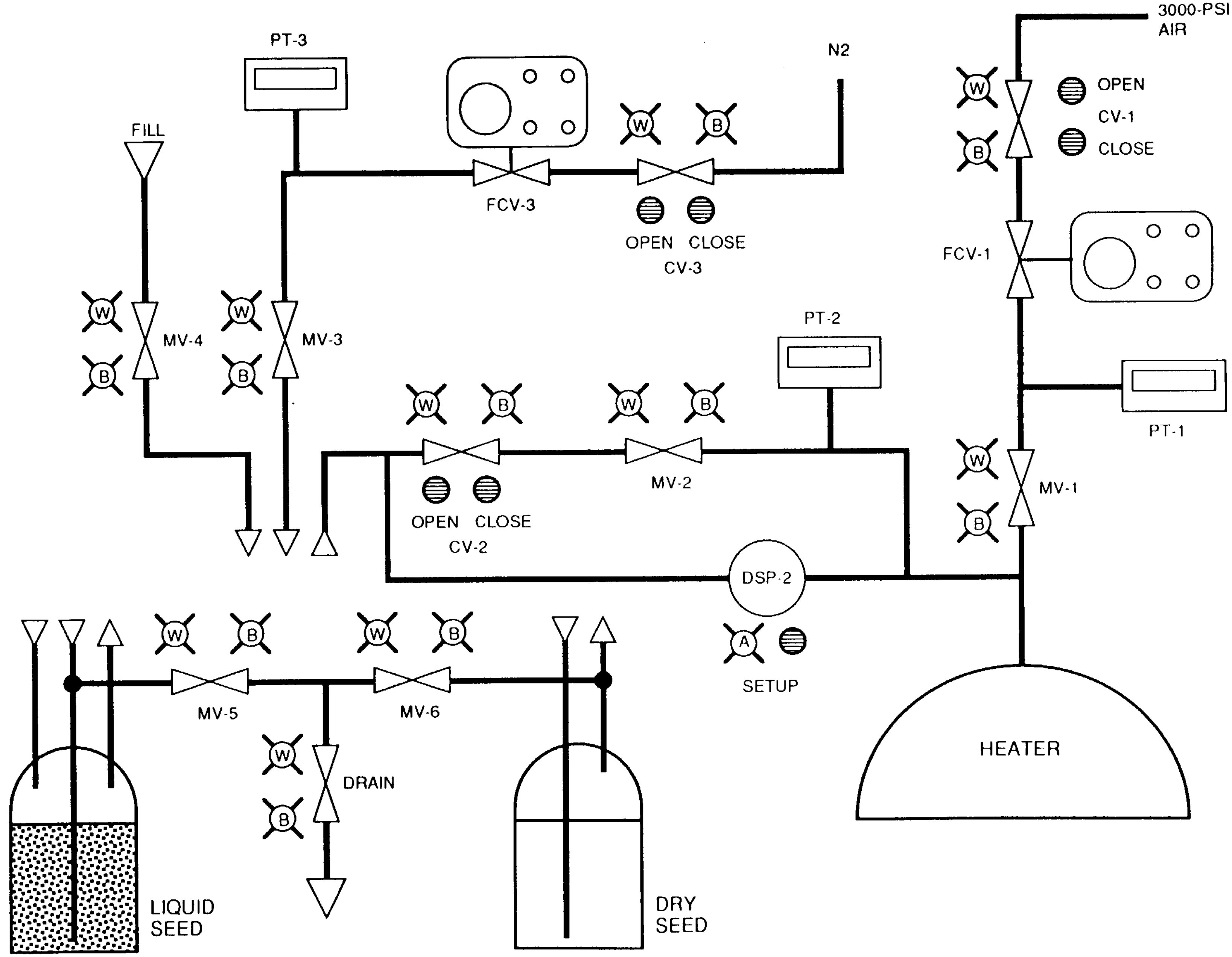
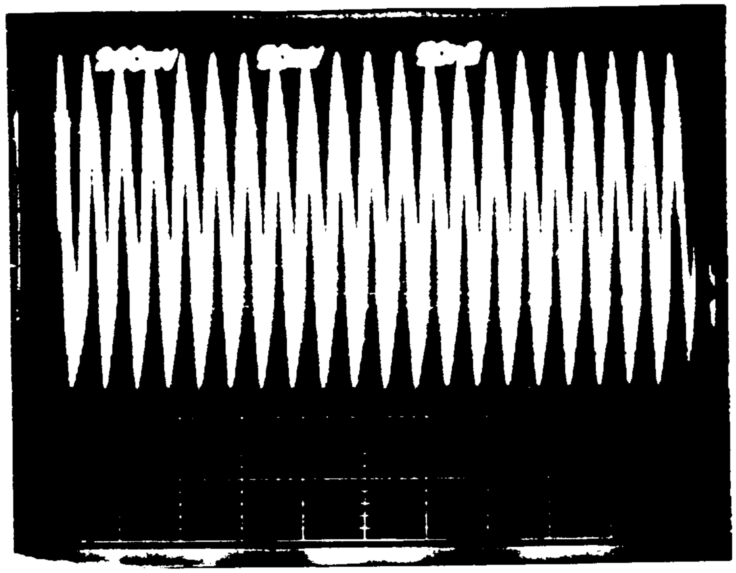
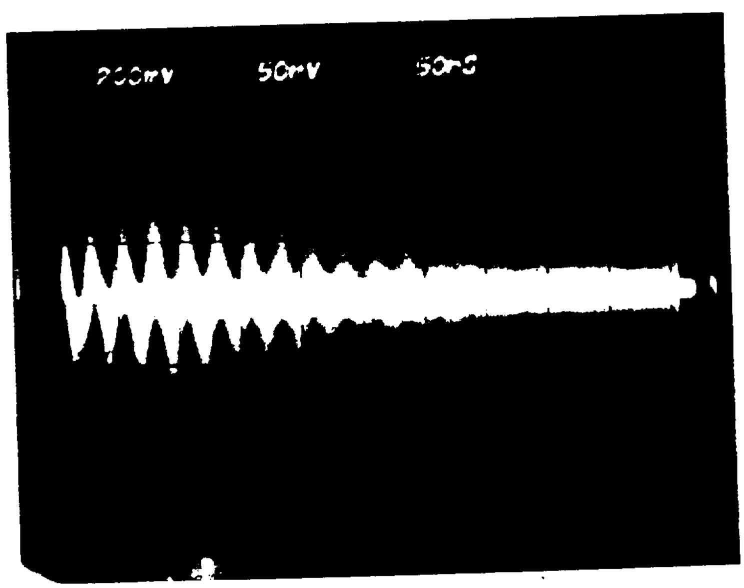


Figure 1. Schematic of the 3.5 ft. HWT LDV Seeder System.



a.) Wind off.



b.) Wind on.

Figure 2. Laser Doppler Velocimeter Signals
(Vertical Velocity Component).

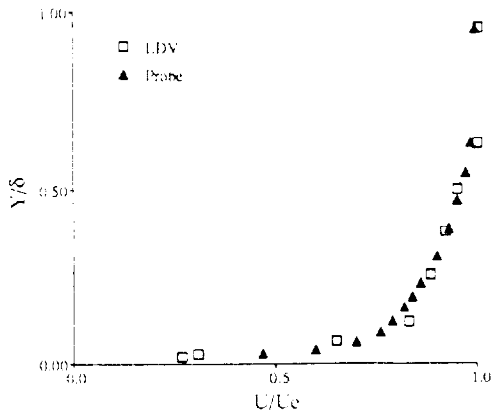


Figure 3. Comparison of Probe and Laser Velocimeter Data.

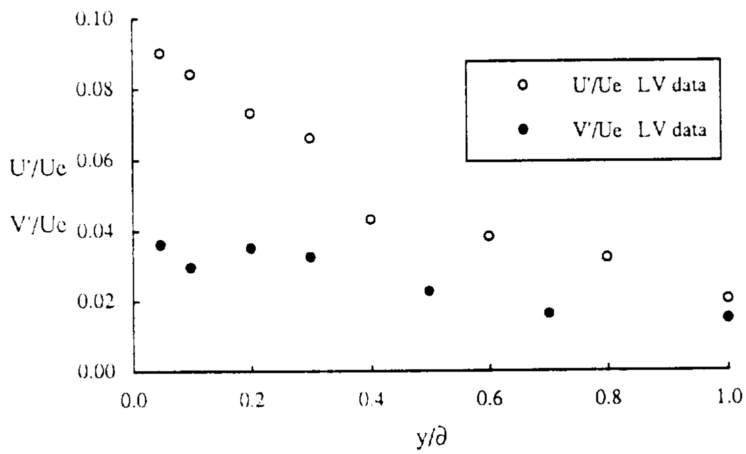


Figure 4. Velocity Fluctuations across the Zero Pressure Gradient Boundary Layer.

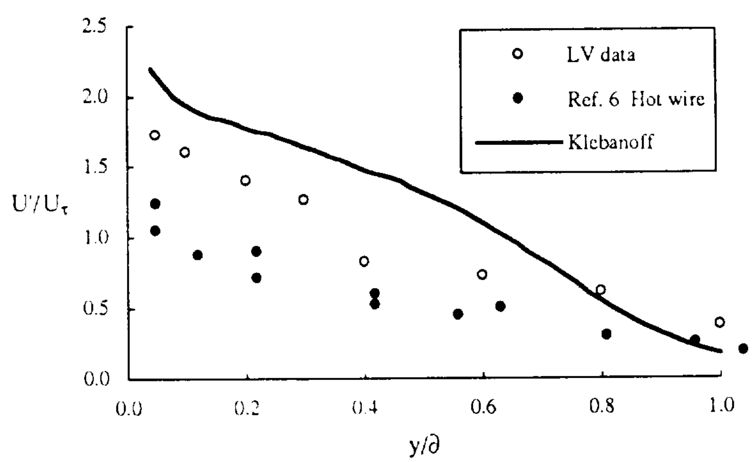


Figure 5. Comparison of Axial Velocity Fluctuations.

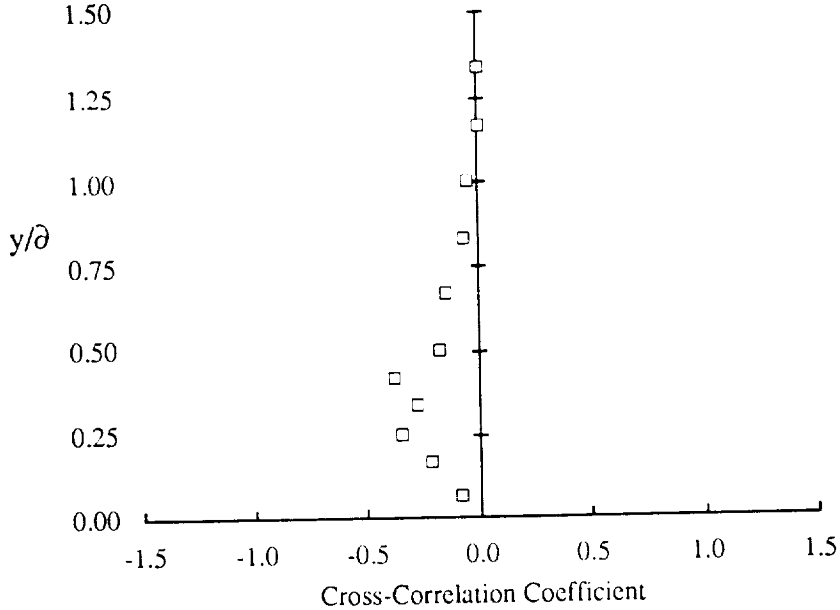


Figure 6. Turbulent Velocity Cross-Correlation Coefficient.

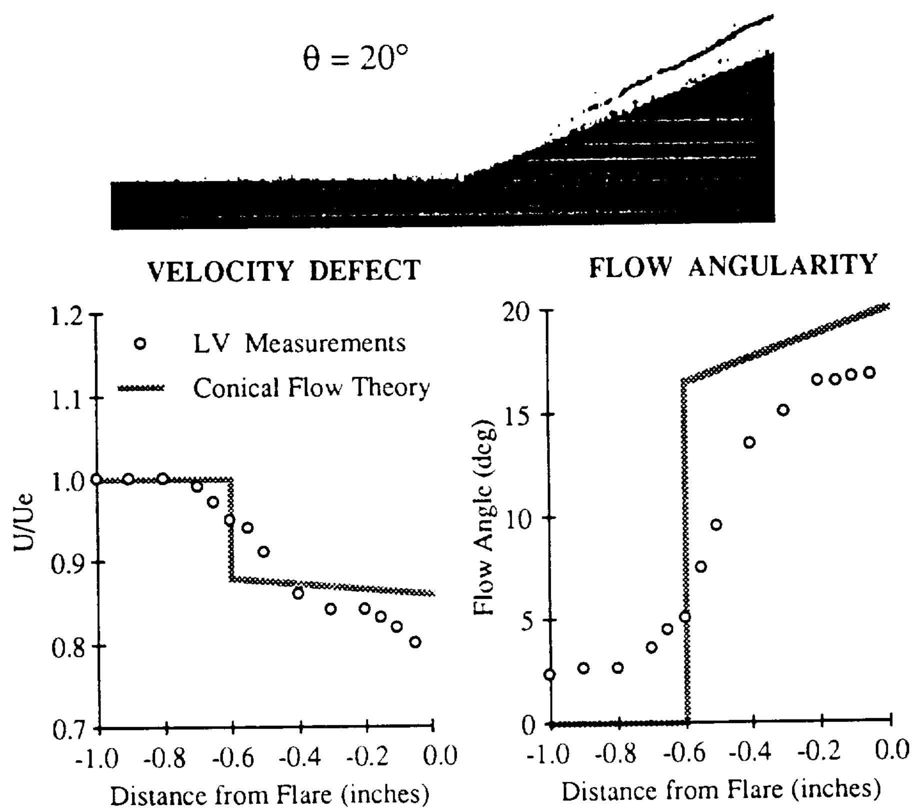


Figure 7. Laser Velocimeter Measurements Across an Oblique Shock Wave.

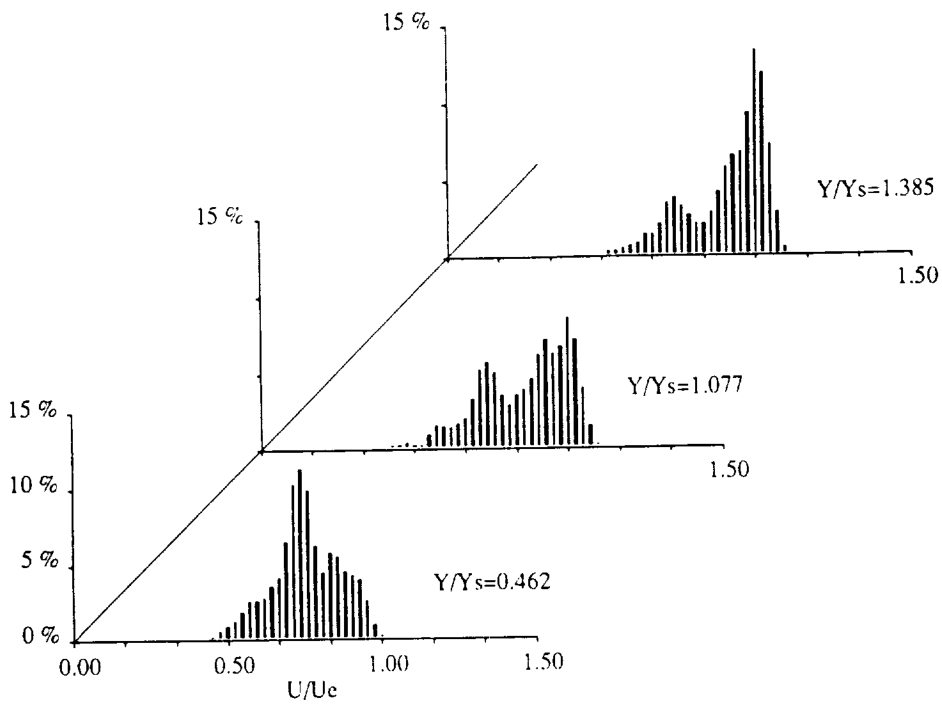


Figure 8. Particle Response in Hypersonic Flow.

Laser Velocimeter Data Acquisition System

to

Sun Sparc Station S11W 16 Bit Parallel Interface

Documentation

Complere Inc.
December 1992

LVDAS to SUN 16 Bit Parallel Interface.

TABLE OF CONTENTS

LIST OF FIGURES

1.0 INTRODUCTION

This documentation describes the LVDAS to SUN interface as well as the data acquisition commands that control the flow of data between the two devices. Section 2 of this documentation provides a detailed schematic drawing of the interface cable, a drawing showing the SUN high density connector pin locations, a drawing showing the LVDAS circular connector pin locations, and timing diagrams for the transfer of data between the two devices.

Section 3 of this documentation provides a detailed description of the data acquisition commands sent to the LVDAS to control the flow of data between the two devices. The types of data, quantity of data, the data acquisition time, and the data formats are also described in Section 3.

The LVDAS can acquire up to 10,000 coincident data sets. Each data sets is composed of 10 words where the word size is 16 bits or 2 bytes. Therefore, the total buffer size is 10,000*10*2 which is equal to 200,000 bytes.

2.0 INTERFACE CABLE

The interface between the Laser Velocimeter Data Acquisition System (LVDAS) and the SUN Sparc Station Computer is a 16 bit parallel general purpose input / output interface. The interface cable shown in Figure 1 consists of a standard cable (SUN EDT Part Number: CAB-A-25) with the terminating connectors on one end removed and replaced with a 55 pin circular connector (Cannon Part Number: MS3470W22-55P). The 80 pin high density connector attaches to the single slot interface card (SUN Part Number: S11W / S16D) within the SUN computer. The pin locations for the high density connector are shown in Figure 2. The 55 pin circular Cannon connector attaches to the Parallel I/O port at the back of the LVDAS. The pin locations of the circular connector are shown in Figure 3.

The timing diagram in Figure 4 shows the handshake sequence for transferring commands or data from the SUN computer to the LVDAS. The timing diagram in Figure 5 shows the handshake sequence for transferring data from the LVDAS to the SUN computer.

2.1 LVDAS to SUN Interface Cable.

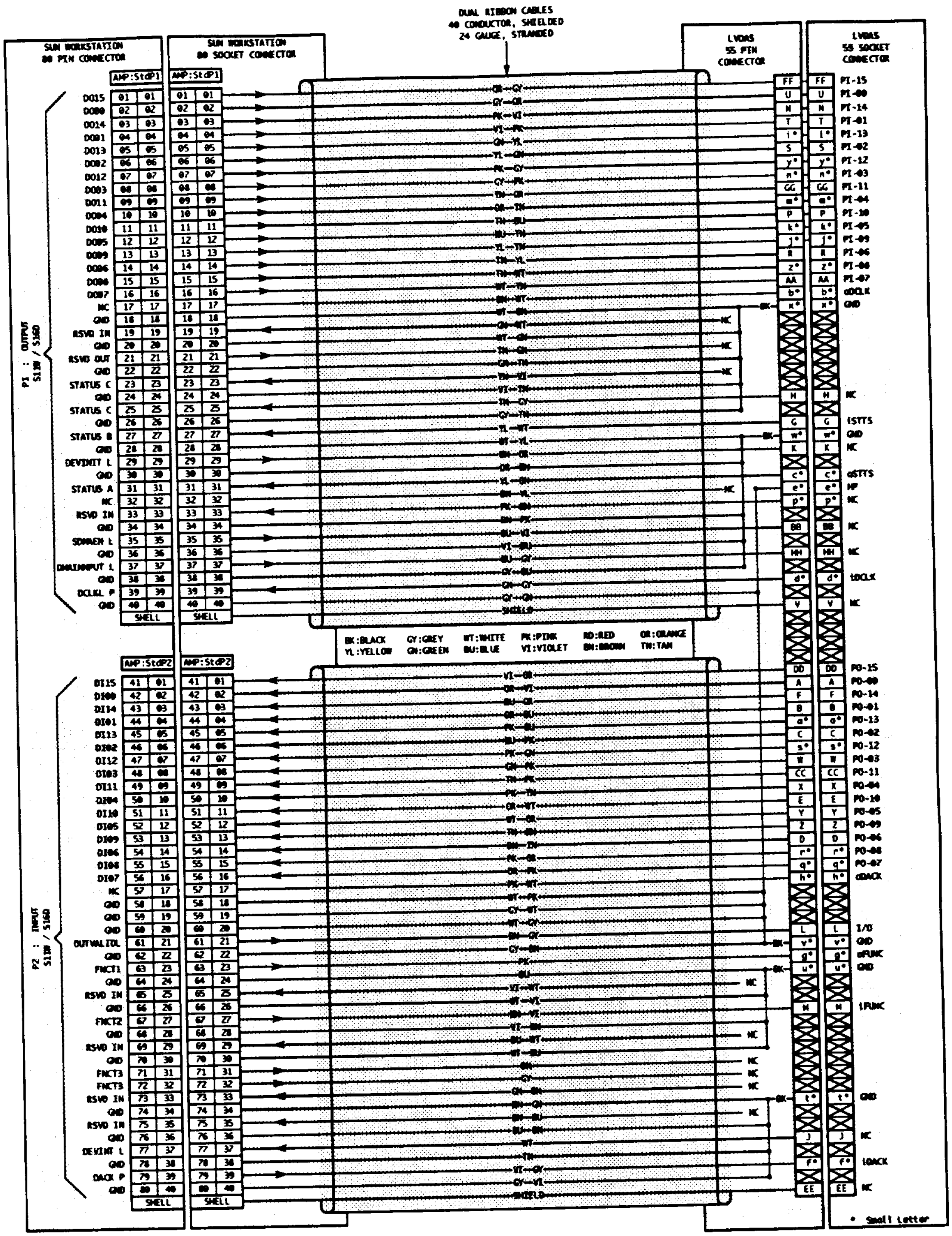


Figure 1. LVDAS to SUN Interface Cable Schematic Drawing.

2.2 SUN High Density Connector.

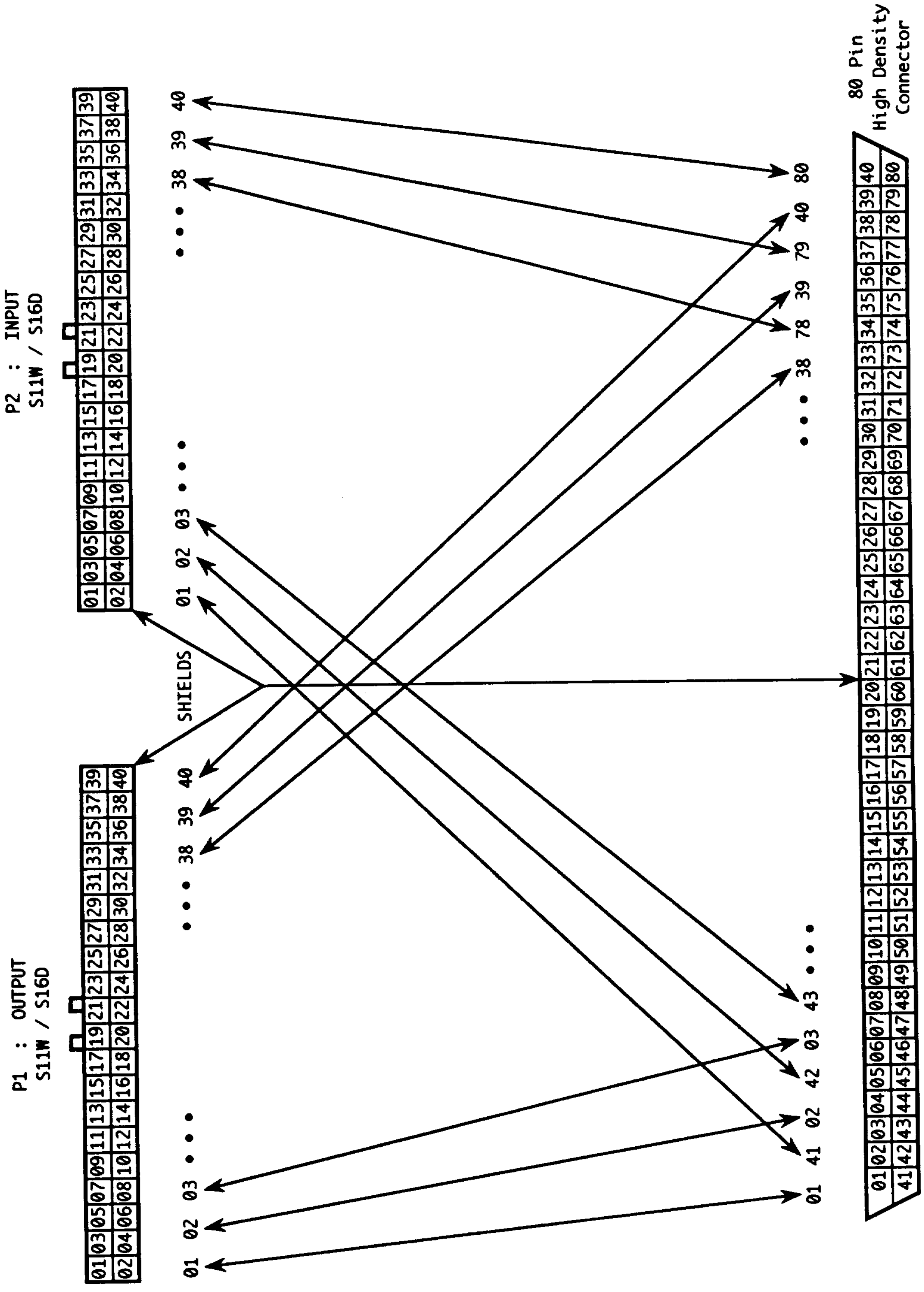
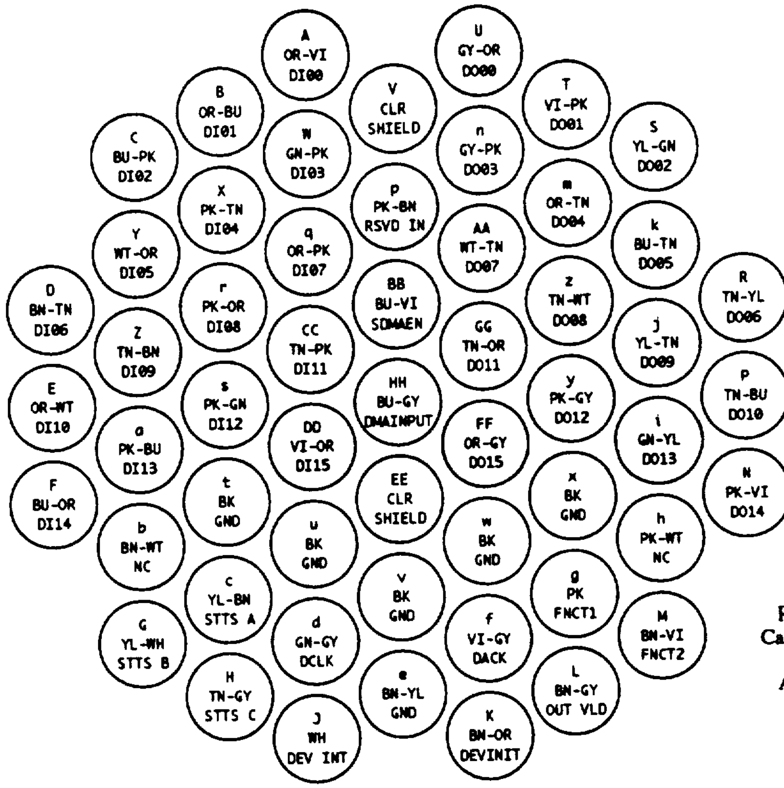
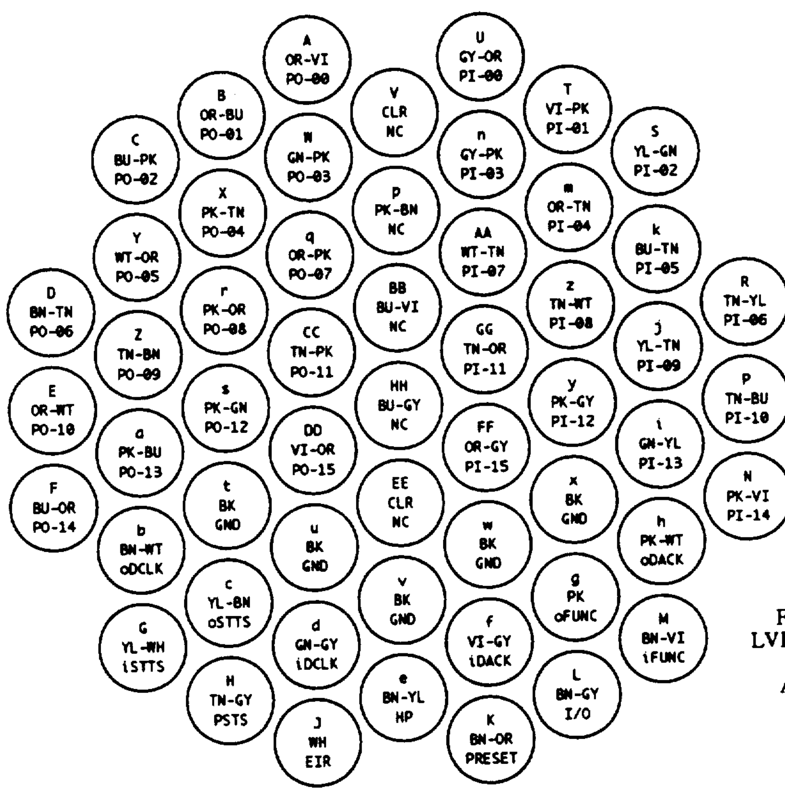


Figure 2. SUN High Density Connector Pin Locations.

2.3 LVDAS Circular Connector.



Rear View of Cable Connector. SUN Pin Assignments.



Front View of LVDAS Connector. LVDAS Pin Assignments.

Figure 3. LVDAS Circular Connector Pin Locations.

2.4 Handshake Timing Diagram for Transfer of Data from SUN to LVDAS.

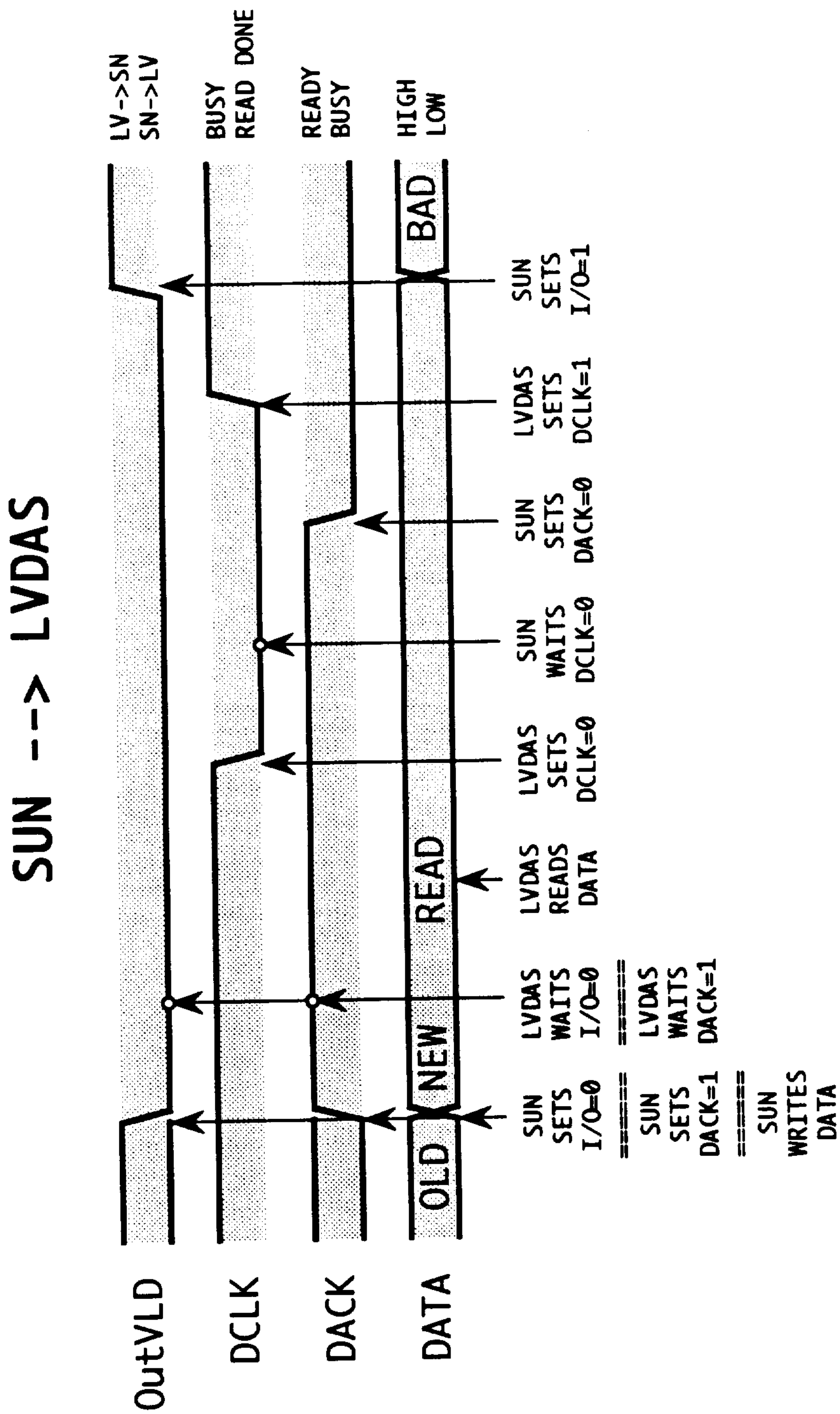


Figure 4. Handshake Timing Diagram for Transfer of Data from SUN Computer to LVDAS.

2.5 Handshake Timing Diagram for Transfer of Data from LVDAS to SUN.

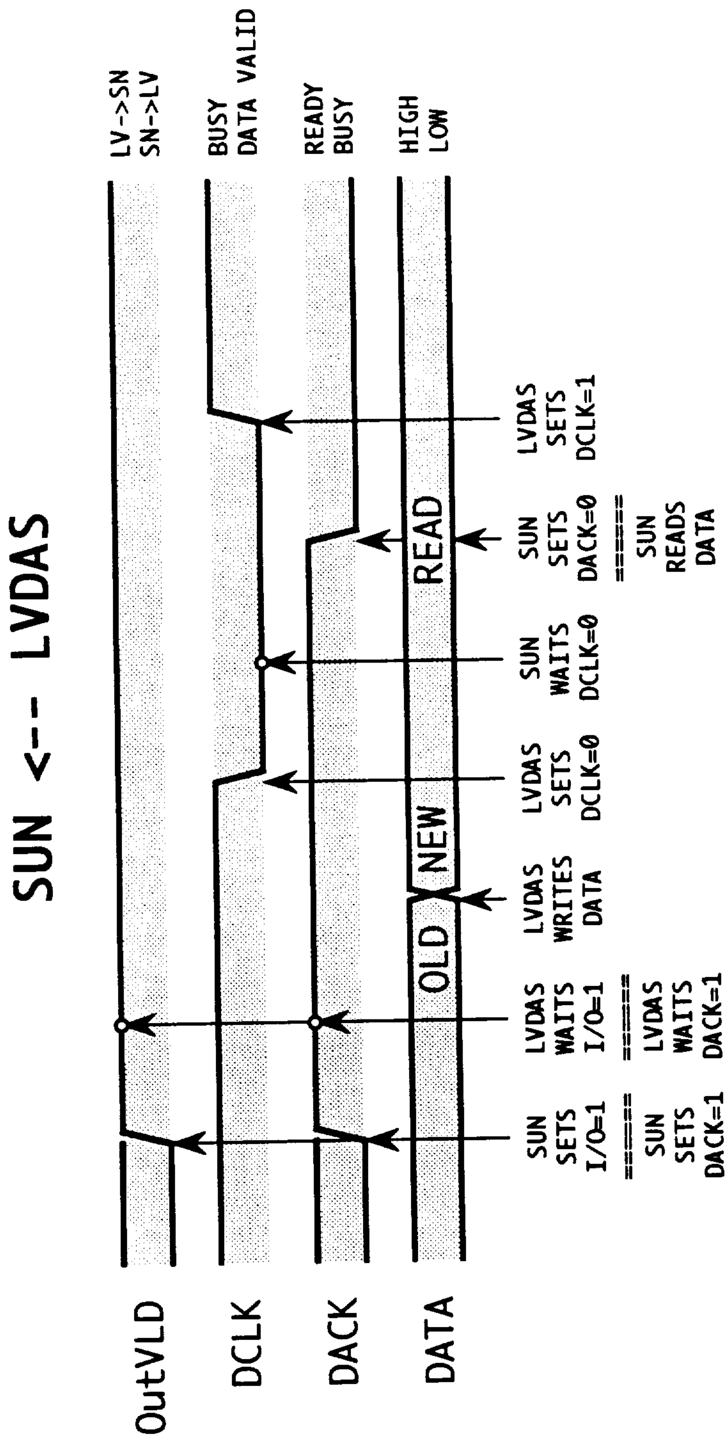


Figure 5. Handshake Timing Diagram for Transfer of Data from LVDAS to SUN Computer.

3.0 DATA ACQUISITION COMMANDS

This section provides a detailed description of the data acquisition commands and parameters sent to the LVDAS to control the flow of data between the two devices.

Commands sent to the LVDAS tell the LVDAS to perform a specific task.

Parameters sent to the LVDAS specify the conditions under which the data acquisition is to take place. Parameters, depending on the command, might include the desired data acquisition time, the desired coincidence time, the inter-arrival and coincidence time exponents, the desired coincidence channel selection, and the desired number of coincident data set samples.

The types of data returned to the computer, depending on the command, might include the inter-arrival time, coincidence time and status, valid data indication, digital frequency data, and digitized analog voltage data.

3.1 "CS" Command: Sample All Channels with Coincidence.

The "CS" command will acquire a finite number of coincident data sets over a finite acquisition time. The following commands, parameters, and data are transferred between the LVDAS and the computer:

WORD	SYMBOL	DESCRIPTION	DIRECTION	LENGTH	TYPE
1	**Cmnd**	"CS" command	Computer to LVDAS	1	Command
2&3	**DAtime**	Desired acquisition time	Computer to LVDAS	2	Parameter
4&5	**DCtime**	Desired coincidence time	Computer to LVDAS	2	Parameter
6	**ATexp**	Inter-arrival time exponent	Computer to LVDAS	1	Parameter
7	**CTexp**	Coincidence time exponent	Computer to LVDAS	1	Parameter
8	**Cmask**	Coincidence mask	Computer to LVDAS	1	Parameter
9	**DNsam**	Desired number of samples	Computer to LVDAS	1	Parameter
10	**RNsam**	Realized number of samples	LVDAS to Computer	1	Parameter
11	**Data0**	Inter-arrival time	LVDAS to Computer	1	Data
12	**Data1**	Coincidence time	LVDAS to Computer	1	Data
13	**Data2**	Coincidence status	LVDAS to Computer	1	Data
14	**Data3**	Not used	LVDAS to Computer	1	Data
15	**Data4**	Data valid	LVDAS to Computer	1	Data
16	**Data5**	Digital channel #1 raw data	LVDAS to Computer	1	Data
17	**Data6**	Digital channel #2 raw data	LVDAS to Computer	1	Data
18	**Data7**	Digital channel #3 raw data	LVDAS to Computer	1	Data
19	**Data8**	Analog channel #1 raw data	LVDAS to Computer	1	Data
20	**Data9**	Not used	LVDAS to Computer	1	Data

The data words 11 through 20 above are repeated **RNsam** times.

The range (min & max), units, and format for the above commands, parameters, and data are shown below:

SYMBOL	MIN	MAX	UNITS	FORMAT
Cmnd	"CS"	-	none	2 ASCII Bytes
DAtime	0	4,294,967,295	100ns	Unsigned 32 bit integer
DCtime	0	4,294,967,295	100ns	Unsigned 32 bit integer
ATexp	0	16	none	Unsigned 16 bit integer
CTexp	0	16	none	Unsigned 16 bit integer
Cmask	1	7	none	Unsigned 16 bit integer
DNsam	0	10,000	none	Unsigned 16 bit integer
RNsam	0	10,000	none	Unsigned 16 bit integer
Data0	0	65,535	ns*	Unsigned 16 bit integer
Data1	0	65,535	ns*	Unsigned 16 bit integer
Data2	0	15	none	Unsigned 16 bit integer
Data3	0	0	none	Unsigned 16 bit integer
Data4	1	1	none	Unsigned 16 bit integer
Data5	0	65,535	Hz*	Unsigned 16 bit integer
Data6	0	65,535	Hz*	Unsigned 16 bit integer
Data7	0	65,535	Hz*	Unsigned 16 bit integer
Data8	-32,768	32,767	volts*	Signed 16 bit integer
Data9	65,535	65,535	none	Signed 16 bit integer

The data words whose units are noted by a * are encoded. Their values in the specified units can be calculated using the raw encoded data.

The command word **Cmnd** (=CS) tells the LVDAS that the computer will want to acquire laser velocimeter data with coincidence. The maximum desired acquisition time **DAtime** and desired coincidence time **DCtime** are specified in 100 ns counts and each is sent to the LVDAS as two 16 bit words concatenated into one 32 bit unsigned integer. For example, counts of 50000, 10000000, and 600000000 would yield times of 5 milliseconds, 1 second, and 1 minute respectively.

The inter-arrival time exponent **ATexp** and coincidence time exponent **CTexp** are use to modify the inter-arrival and coincidence times. The LVDAS measures these times with a resolution of 100 ns and an unsigned integer data size of 32 bits. The 32 bit inter-arrival time is shifted right by the number of bits specified by the inter-arrival time exponent **ATexp**. The 32 bit coincidence time is shifted right by the number of bits specified by the coincidence time exponent **CTexp**. The resulting 16 bit words are later sent to the computer.

The coincidence mask **Cmask** determines the desired coincidence criterion. The least significant three bits individually select the digital channels #1, #2, and #3 for coincidence. Valid coincidence masks are as follows:

Coincidence Mask (decimal)	Coincidence Mask (binary)	Channel #3 (selected)	Channel #2 (selected)	Channel #1 (selected)
0	0000 0000 0000 0000	NO	NO	NO
1	0000 0000 0000 0001	NO	NO	YES
2	0000 0000 0000 0010	NO	YES	NO
3	0000 0000 0000 0011	NO	YES	YES
4	0000 0000 0000 0100	YES	NO	NO
5	0000 0000 0000 0101	YES	NO	YES
6	0000 0000 0000 0110	YES	YES	NO
7	0000 0000 0000 0111	YES	YES	YES

The desired number of samples **DNsam** specifies the number of coincident data sets to be acquired within the previously specified desired data acquisition time **DAtime**. The data acquisition commences when **DNsam** is received by the LVDAS.

The data acquisition terminates when one of two conditions occur. The first terminating condition is that **DNsam** coincident data sets are realized before the allocated data acquisition time **DAtime** expires. In this case, the desired **DNsam** and realized **RNsam** number of samples are the same. The second terminating condition is that **DNsam** coincident data sets are <u>not</u> realized before the allocated data acquisition time **DAtime** expires. In this case, the realized number of samples **RNsam** may be less than the desired number of samples **DNsam**. In both terminating conditions, this value (**RNsam**) is then sent from the LVDAS to the computer to indicate data acquisition completion and to also indicate the size of the data array to be subsequently transferred to the computer.

Each coincident data set consists of ten 16 bit words. **RNsam** indicates the number of acquired coincident data sets. Therefore, there will be 10***RNsam** words sent from the LVDAS to the computer. The computer's data array should be dimensioned accordingly. The 10 words will include the inter-arrival and coincidence times, the coincidence status and data valid words, as well as the digital and analog raw data words.

The inter-arrival time **Data0** and coincidence time **Data1** raw data words can be converted to the actual inter-arrival time **IAtime** and realized coincidence time **RCtime** in seconds using the following equations:

IAtime = **Data0** * (2^**ATexp**) / (10^7) seconds

RCtime = **Data1** * (2^**CTexp**) / (10^7) seconds

The coincidence status **Status** and data valid **Valid** words, **Data2** and **Data3** respectively, indicate the channels that have new data in the data set and the validity of the data. If **Valid**=0 then the data set does not contain valid data. If **Valid**=1 then the data set does contain valid data. The least significant four **Status** bits individually indicate weather

of not new data has been acquired on the digital and analog channels:

Status Word (decimal)	Status Word (binary)	Analog Ch #1 (new)	Digital Ch #3 (new)	Digital Ch #2 (new)	Digital Ch #1 (new)
0	0000 0000 0000 0000	NO	NO	NO	NO
1	0000 0000 0000 0001	NO	NO	NO	YES
2	0000 0000 0000 0010	NO	NO	YES	NO
3	0000 0000 0000 0011	NO	NO	YES	YES
4	0000 0000 0000 0100	NO	YES	NO	NO
5	0000 0000 0000 0101	NO	YES	NO	YES
6	0000 0000 0000 0110	NO	YES	YES	NO
7	0000 0000 0000 0111	NO	YES	YES	YES
8	0000 0000 0000 1000	YES	NO	NO	NO
9	0000 0000 0000 1001	YES	NO	NO	YES
10	0000 0000 0000 1010	YES	NO	YES	NO
11	0000 0000 0000 1011	YES	NO	YES	YES
12	0000 0000 0000 1100	YES	YES	NO	NO
13	0000 0000 0000 1101	YES	YES	NO	YES
14	0000 0000 0000 1110	YES	YES	YES	NO
15	0000 0000 0000 1111	YES	YES	YES	YES

The raw data words **Data5**, **Data6**, and **Data7** contain the digital data from the Macrodyne laser velocimeter counter signal processors. These digital data can be converted into frequencies using the following equations:

Mantissa1 = Bits 0 to 9 of **Data5**
Mantissa2 = Bits 0 to 9 of **Data6**
Mantissa3 = Bits 0 to 9 of **Data7**

Exponent1 = Bits 10 to 13 of **Data5**
Exponent2 = Bits 10 to 13 of **Data6**
Exponent3 = Bits 10 to 13 of **Data7**

Fringes1 : If bit 14 of **Data5**=0 then **Fringes1**=16 else **Fringes1**=8
Fringes2 : If bit 14 of **Data6**=0 then **Fringes2**=16 else **Fringes2**=8
Fringes3 : If bit 14 of **Data7**=0 then **Fringes3**=16 else **Fringes3**=8

Period1 = **Mantissa1** * (2^**Exponent1**) / (10^9) (seconds)
Period2 = **Mantissa2** * (2^**Exponent2**) / (10^9) (seconds)
Period3 = **Mantissa3** * (2^**Exponent3**) / (10^9) (seconds)

Frequency1 = **Fringes1 / Period1** (Hz)
Frequency2 = **Fringes2 / Period2** (Hz)
Frequency3 = **Fringes3 / Period3** (Hz)

The following equation is used to convert the raw data word **Data8** into a voltage:

Analog = **Data8** * 5 / 32768 (volts)

3.2 "SC" Command: Sample One Channel.

The "SC" command will acquire 1000 data samples from one channel. The following commands, parameters, and data are transferred between the LVDAS and the computer:

WORD	SYMBOL	DESCRIPTION	DIRECTION	LENGTH	TYPE
1	**Cmnd1**	"DT" command	Computer to LVDAS	1	Command
2	**Cmnd2**	"SC" command	Computer to LVDAS	1	Command
3	**Channel**	Channel Number	Computer to LVDAS	1	Parameter
4	**Cmnd3**	"ET" command	Computer to LVDAS	1	Command
5	**Cmnd4**	"RM" command	Computer to LVDAS	1	Command
6&7	**First**	Memory location	Computer to LVDAS	2	Parameter
8&9	**Last**	Memory location	Computer to LVDAS	2	Parameter
10&11*	**Data1**	Inter-arrival time	LVDAS to Computer	2	Data
12*	**Data2**	Channel number	LVDAS to Computer	1	Data
13*	**Data3**	Channel data	LVDAS to Computer	1	Data

The data words 10 through 13 are repeated 1000 times.

The range (min & max), units, and format for the above commands, parameters, and data are shown below:

SYMBOL	MIN	MAX	UNITS	FORMAT
Cmnd1	"DT"	-	none	2 ASCII Bytes
Cmnd2	"SC"	-	none	2 ASCII Bytes
Channel	1	7	none	Unsigned 16 bit integer
Cmnd3	"ET"	-	none	2 ASCII Bytes
Cmnd4	"RM"	-	none	2 ASCII Bytes
First	`08F00000` hex	-	none	Unsigned 32 bit integer
Last	`08F01F3F` hex	-	none	Unsigned 32 bit integer
Data1	0	4,294,967,295	100ns	Unsigned 32 bit integer
Data2	0	6	none	Unsigned 16 bit integer
Data3	see text	see text	see text	see text

The first command word **Cmnd1** (=DT) tells the LVDAS to disable internal timers which temporarily stops updating of the front panel displays. Sending out **Cmnd1** is optional. The second command word **Cmnd2** (=SC) tells the LVDAS that the computer will want to acquire laser velocimeter or analog data on one channel only. The data word **Channel** specifies the channel number for which data will be acquired. Valid channel

numbers are as follows:

Channel Number	Channel Description	Generates Data Word	Generates Inter-Arrival Time Words
1	Digital channel #1	YES	YES
2	Digital channel #2	YES	YES
3	Digital channel #3	YES	YES
4	Analog channel #1	YES	YES
6	External trigger timer	NO	YES
7	Inter-arrival time timer	NO	YES

The data acquisition commences when **Channel** is received by the LVDAS. The third command word **Cmnd3** (=ET) tells the LVDAS to enable internal timers which activates the updating of the front panel displays. After 1000 data samples have been acquired, then the third command word **Cmnd3** will be executed. The computer can now read back the data from the buffer's memory. Reading memory is initiated by sending the forth command word **Cmnd4** (=RM) and the two memory buffer parameters **First** and **Last**.

The LVDAS will respond by sending 4 words of data per sample to the computer. The first 2 words in **Data1** contain the inter-arrival time **IAtime**; the third word in **Data2** contains the channel number **Channel**; and the forth word in **Data3** contains the channel's data.

The two inter-arrival time raw data words in **Data1** can be converted to the actual inter-arrival time **IAtime** in seconds using the following equation:

IAtime = **Data1** / (10^7) seconds

The type of data, its range (min & max), units, and format returned in **Data3** depend on which channel, specified by **Channel**, the data was acquired on. (Note: The LVDAS will return channel numbers minus one: 0..6; not 1..7).

Channel Number	Channel Description	Generates Data Word	Generates Inter-Arrival Time Words
0	Digital Channel #1	YES	YES
1	Digital Channel #2	YES	YES
2	Digital Channel #3	YES	YES
3	Analog Channel #1	YES	YES
5	External Trigger Timer	NO	YES
6	Inter-Arrival Time Timer	NO	YES

CHANNEL	MIN	MAX	UNITS	FORMAT
0	0	65,535	Hz*	Unsigned 16 bit integer
1	0	65,535	Hz*	Unsigned 16 bit integer
2	0	65,535	Hz*	Unsigned 16 bit integer
3	-32,768	32,767	volts*	Signed 16 bit integer

The data words whose units are noted by a * are encoded. Their values in the specified units can be calculated using the raw encoded data.

If the data was acquired on channels 0 through 2, then the following equations should be used to convert the digital data from the Macrodyne laser velocimeter counter signal processors into frequencies:

Mantissa = Bits 0 to 9 of **Data3**
Exponent = Bits 10 to 13 of **Data3**
Fringes : If bit 14 of **Data3**=0 then **Fringes**=16 else **Fringes**=8
Period = **Mantissa** * (2^**Exponent**) / (10^9) (seconds)
Frequency = **Fringes / Period** (Hz)

If the data was acquired on channel 3, then the following equation should be used to convert the raw data word into a voltage:

Analog = **Data3** * 5 / 32768 (volts)

Channels 4 through 6 produce an inter-arrival time but do not generate any meaningful data. Their data is ignored.

3.3 "SA" Command: Sample All Channel.

The "SA" command will acquire 1000 data samples from all channels. The 1000 samples will be spread out over all enabled channels. Channels with higher data rates will generate more samples than channels with lower data rates. The sum total of all samples will be 1000 samples. The following commands, parameters, and data are transferred between the LVDAS and the computer:

WORD	SYMBOL	DESCRIPTION	DIRECTION	LENGTH	TYPE
1	**Cmnd1**	"DT" command	Computer to LVDAS	1	Command
2	**Cmnd2**	"SA" command	Computer to LVDAS	1	Command
3	**Mask**	Channel Mask	Computer to LVDAS	1	Parameter
4	**Cmnd3**	"ET" command	Computer to LVDAS	1	Command
5	**Cmnd4**	"RM" command	Computer to LVDAS	1	Command
6&7	**First**	Memory location	Computer to LVDAS	2	Parameter
8&9	**Last**	Memory location	Computer to LVDAS	2	Parameter
10&11*	**Data1**	Inter-arrival time	LVDAS to Computer	2	Data
12*	**Data2**	Channel number	LVDAS to Computer	1	Data
13*	**Data3**	Channel data	LVDAS to Computer	1	Data

The data words 10 through 13 are repeated 1000 times.

The range (min & max), units, and format for the above commands, parameters, and data are shown below:

SYMBOL	MIN	MAX	UNITS	FORMAT
Cmnd1	"DT"	-	none	2 ASCII Bytes
Cmnd2	"SA"	-	none	2 ASCII Bytes
Mask	1	127	none	Unsigned 16 bit integer
Cmnd3	"ET"	-	none	2 ASCII Bytes
Cmnd4	"RM"	-	none	2 ASCII Bytes
First	08F00000 hex	-	none	Unsigned 32 bit integer
Last	08F01F3F hex	-	none	Unsigned 32 bit integer
Data1	0	4,294,967,295	100ns	Unsigned 32 bit integer
Data2	0	6	none	Unsigned 16 bit integer
Data3	see text	see text	see text	see text

The first command word **Cmnd1** (=DT) tells the LVDAS to disable internal timers which temporarily stops updating of the front panel displays. Sending out **Cmnd1** is

optional. The second command word **Cmnd2** (=SA) tells the LVDAS that the computer will want to acquire laser velocimeter and/or analog data on all channels. The data word **Mask** specifies the channel numbers for which data will be acquired. Each bit in the **Mask** enable data acquisition on the relevant channels. Channels whose **Mask** bit equals zero will be ignored. Channels whose **Mask** bit equals one will be serviced each time data becomes available.

Channel Number	Mask Bit	Channel Description	Generates Data Word	Generates Inter-Arrival Time Words
1	0	Digital channel #1	YES	YES
2	1	Digital channel #2	YES	YES
3	2	Digital channel #3	YES	YES
4	3	Analog channel #1	YES	YES
6	5	External trigger timer	NO	YES
7	6	Inter-arrival time timer	NO	YES

The data acquisition commences when **Mask** is received by the LVDAS. The third command word **Cmnd3** (=ET) tells the LVDAS to enable internal timers which activates the updating of the front panel displays. After 1000 data samples have been acquired, then the third command word **Cmnd3** will be executed. The computer can now read back the data from the buffer's memory. Reading memory is initiated by sending the forth command word **Cmnd4** (=RM) and the two memory buffer parameters **First** and **Last**.

The LVDAS will respond by sending 4 words of data per sample to the computer. The first 2 words in **Data1** contain the channel inter-arrival time **IAtime**; the third word in **Data2** contains the channel number **Channel**; and the forth word in **Data3** contains the channel's data.

The channel inter-arrival times are the inter-arrival times of data samples acquired on the same channel. The average channel inter-arrival time for all samples acquired on a specific channel yield that channels data rate (rate=1/period). The two channel inter-arrival time raw data words in **Data1** can be converted to the actual inter-arrival time **IAtime** in seconds using the following equation:

IAtime = **Data1** / (10^7) seconds

The type of data, its range (min & max), units, and format returned in **Data3** depend on which channel, specified by **Channel**, the data was acquired on. (Note: The LVDAS will return channel numbers minus one: 0..6; not 1..7).

Channel Number	Channel Description	Generates Data Word	Generates Inter-Arrival Time Words
0	Digital Channel #1	YES	YES
1	Digital Channel #2	YES	YES
2	Digital Channel #3	YES	YES
3	Analog Channel #1	YES	YES
5	External Trigger Timer	NO	YES
6	Inter-Arrival Time Timer	NO	YES

CHANNEL	MIN	MAX	UNITS	FORMAT
0	0	65,535	Hz*	Unsigned 16 bit integer
1	0	65,535	Hz*	Unsigned 16 bit integer
2	0	65,535	Hz*	Unsigned 16 bit integer
3	-32,768	32,767	volts*	Signed 16 bit integer

The data words whose units are noted by a * are encoded. Their values in the specified units can be calculated using the raw encoded data.

If the data was acquired on channels 0 through 2, then the following equations should be used to convert the digital data from the Macrodyne laser velocimeter counter signal processors into frequencies:

Mantissa = Bits 0 to 9 of **Data3**
Exponent = Bits 10 to 13 of **Data3**
Fringes : If bit 14 of **Data3**=0 then **Fringes**=16 else **Fringes**=8
Period = **Mantissa** * (2^**Exponent**) / (10^9) (seconds)
Frequency = **Fringes / Period** (Hz)

If the data was acquired on channel 3, then the following equation should be used to convert the raw data word into a voltage:

Analog = **Data3** * 5 / 32768 (volts)

Channels 4 through 6 produce an inter-arrival time but do not generate any meaningful data. Their data is ignored.

THE NASA AMES
3.5-FT. HYPERSONIC WIND TUNNEL
LASER VELOCIMETER SYSTEM

CONTRACT REPORT
92-0401

COMPLERE INC.
P.O. BOX 1697
PALO ALTO, CA
APRIL 1992

TABLE OF CONTENTS

LIST OF APPENDICES

SBIR RIGHTS NOTICE

CHAPTER 1

3.5 FT HWT OPTICAL SYSTEM.

CHAPTER 1

3.5 FT HWT Optical System

TABLE OF CONTENTS

LIST OF FIGURES

1.0 THE LASER DOPPLER VELOCIMETER

The layout of the 3.5 FT HWT Laser Doppler Velocimeter is shown schematically in Fig. 1. Details of the plenum optics are shown in Fig. 2.

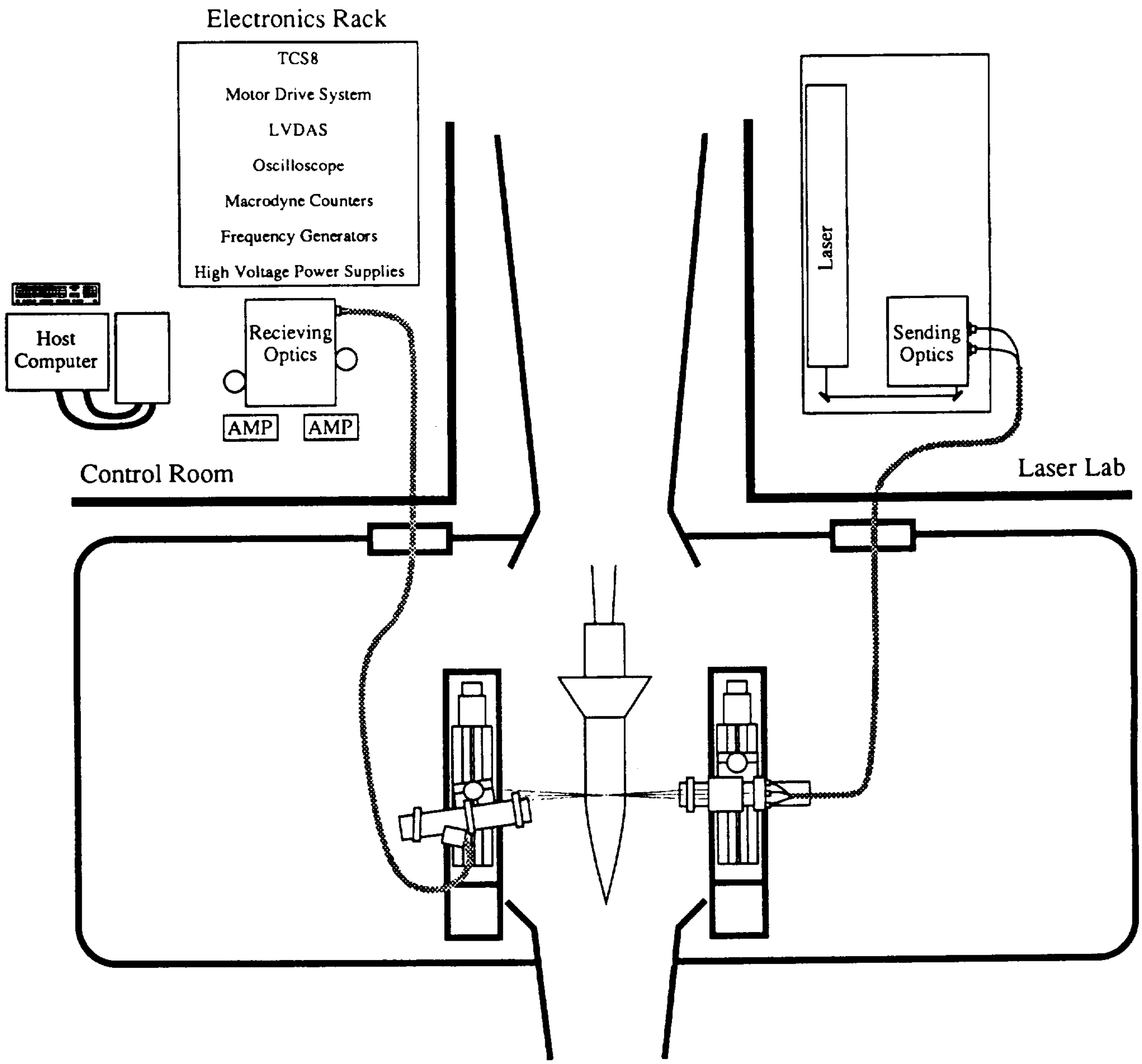


Figure 1 The 3.5 FT HWT 2-D LDV System.

Mean velocity and turbulence measurements are made with a dual-beam velocimeter utilizing a Bragg cell that enables moving interference fringes to be generated in the focal volume so that instantaneous velocity magnitude and direction measurements can be achieved from the frequency shift (f_D) around the incident and modulated laser beam interference frequency (f_0). i.e. $U = \lambda(f_D - f_0) / 2 \sin(\varnothing/2)$ where λ is the wavelength of the incident laser light.

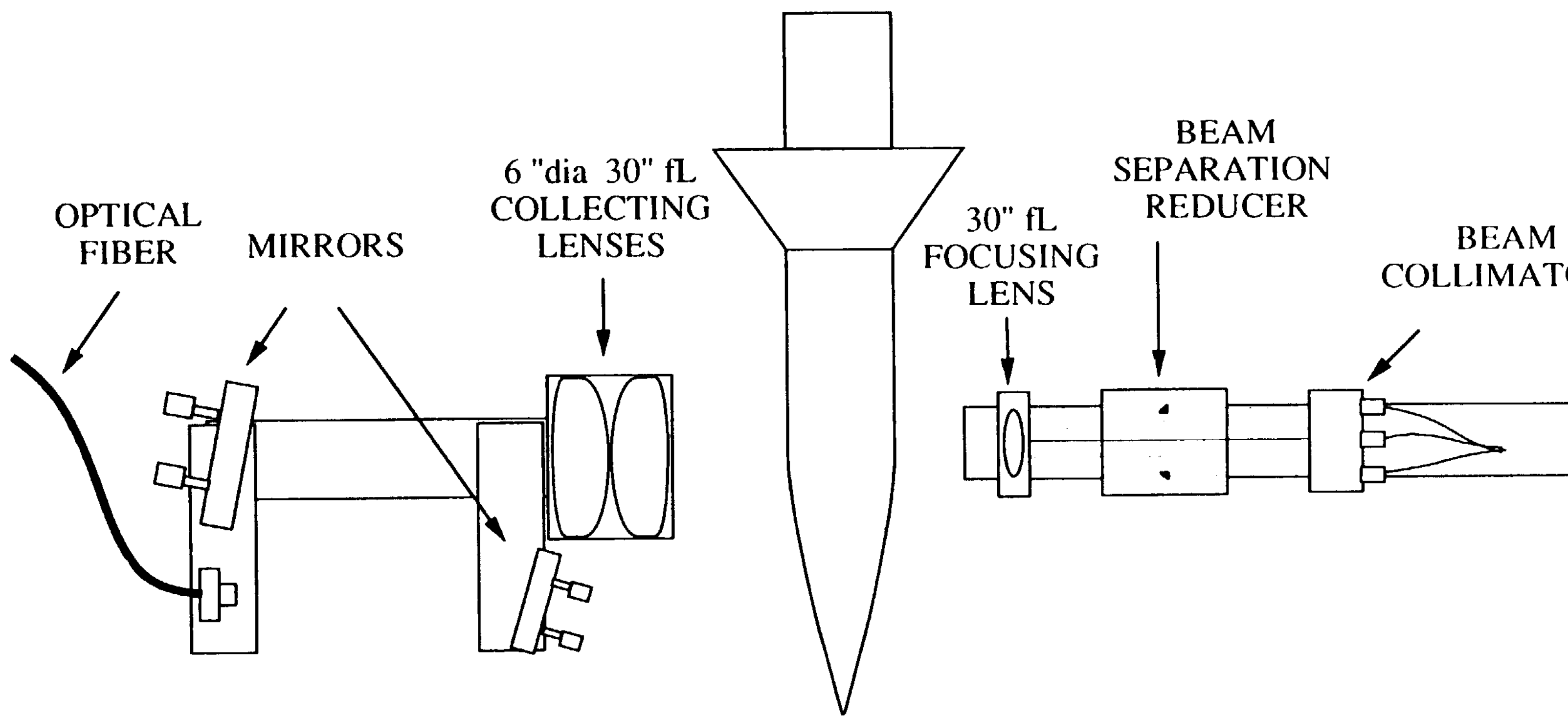


Figure 2. Plenum Optics.

The transmitting optics color separation system (Fig. 3) is straightforward with a few unique features addressing the common problem of beam distortion or thermal blooming at higher laser powers. Frequency shifting is done before the color separation prisms, using a single acousto-optic modulator made of a selected flint glass, which can handle substantial laser power with minimal distortion. This is followed by color separation prisms, the first of which are made of fused silica for power handling capacity. A final prism of dense flint provides maximum angular displacement once the light has been dispersed into numerous beams. Final color selection is made using right angle prisms. The lines used for this application were 514.5 nm and 488 nm. Other laser lines could have been selected.

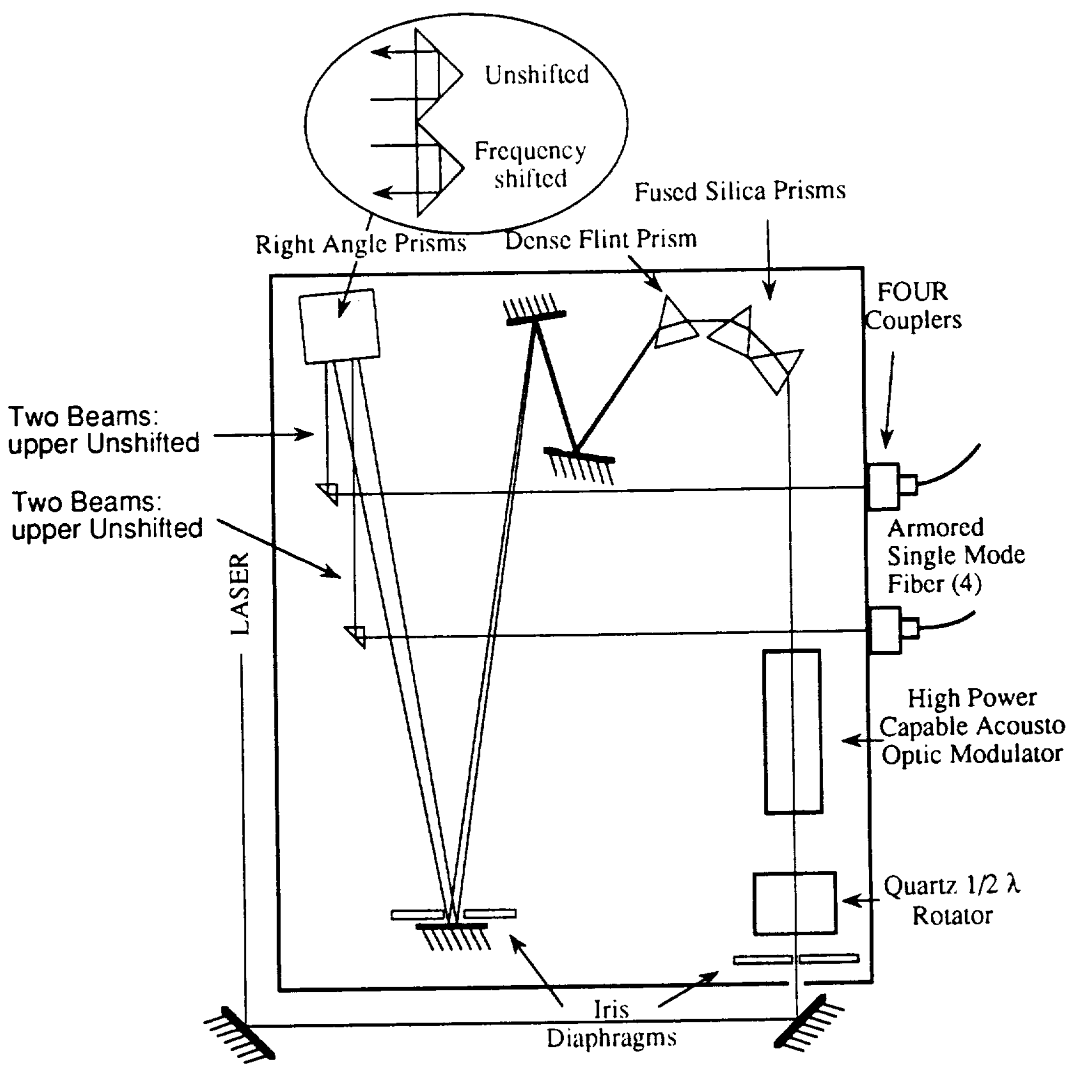


Figure 3 Transmitting Optics Separation System.

Pure fused-silica core single-mode polarization-preserving fibers are used for light transmission; two fibers per color. The use of optical fibers avoids the tedium of mirror-traverse

alignment. The pure fused silica core fibers are less susceptible to the progressive transmission losses which are found in other fibers. Polarization preserving fibers provide greater modal stability when the fibers are flexed or manipulated. For mechanical and thermal protection, the fibers are armored and contained within a conduit which is air cooled within the plenum. Upon exiting the fibers the light is collimated at 2.2 mm dia. with a separation of 60 mm. Adjustable rhomboid prisms reduce the beam separation to .3125 inches. The final focusing lens is 50.8 mm diameter and 750 mm focal length.

Forward-scattered light is collected with a 6 inch diameter, 30 inch focal length lens and focused into a 600 μm multi-mode optical fiber, which conducts it to the color separation and signal detection box through an air cooled conduit. For maximum throughput efficiency of the collected light color separator, a prism separation scheme is used rather than di-chroic filter and interference filters..

Experience has shown that accurate positioning is vital to a successful test program. Position is maintained by a custom designed eight axis capable traverse controller with micro-stepping drives, optical encoder feedback, and limit switch safety stops. Chapter 4 contains a detailed description of the traverse control system.

2.0 TRANSMITTING COLOR SEPARATION SYSTEM ALIGNMENT

2.1 Optics Enclosure

In order to steer the laser beam into the transmitting color separation box (Fig. 3), the best approach is often to use two steering mirrors between the laser and the box. This allows the beam to be fully manipulated without moving the laser or the box.

WARNING

LASER SHOULD BE OPERATED AT MINIMUM POWER DURING ALIGNMENT.

DO NOT STARE AT THE BEAM OR DIFFUSE REFLECTIONS.

WEAR APPROPRIATE PROTECTIVE EYEWEAR.

PROJECT BEAMS ONTO A DULL, FLAT BLACK, NON-FLAMMABLE SURFACE.

Select a position for the color separation box on the optical table which supports the laser and bolt down the box baseplate using the clearance holes at the center of each side of the baseplate.

2.2 Outside Steering Mirrors

Move the polarization rotator and Bragg cell out of the way and open the iris diaphragm fully. Adjust the position and orientation of the input steering mirrors to direct the beam into the color separation box at a height of 4.25 inches from the top surface of the baseplate and 1.25 inches from the inside surface of the front plate. This can be checked by placing transparent rulers into the beam path at each end of the optics box.

2.3 Polarization Rotator

Put the polarization rotator into its holder and adjust it so that the beam travels through the center of the aperture. The polarization will be set after the Brewster angle dispersion prisms are put into position.

2.4 Acousto-Optic Modulator (Bragg Cell)

Set the precision micrometer adjustment of the the Bragg cell mount to the middle of its travel; about three full turns from the stops.

Slide the Bragg cell into the beam.

Connect the Bragg cell RF input to the inside front panel BNC feed through with a short length of RG-58 cable.

Connect the Bragg cell driver to the outside front panel BNC connector.

Loosen the gross movement set screw and position the Bragg cell so that the beam travels through the center of both the input and output apertures. Tighten the gross movement set screw.

Switch on the Bragg cell driver and turn up the drive power so that the diffracted beams can be seen.

Tilt the Bragg cell up and down to identify the first order diffracted beam. The unshifted beam is the one which remains when the Bragg cell driver is switched off. As the Bragg cell is tilted back and forth, the first order diffracted beam will appear above and then below the undiffracted beam. The Bragg cell should be set so that the undiffracted beam is on the bottom. Using the precision adjustment knob, set the Bragg cell tilt to put the maximum amount of light into the upper, first diffracted beam. A laser power meter can be used for this.

Adjust the Bragg cell drive power so that equal power is in both beams. Again, use a laser power meter for the greatest precision in this adjustment. This adjustment should be checked again at the measurement volume after the transmitting optics are completely set up. Coupling efficiency will vary from fiber to fiber. Also, the percentage of laser power diffracted into the shifted beam is wavelength dependent. Bragg cell drive power should be set to the best compromise, remembering that the ideal is equal power in each beam of each pair.

2.5 Dispersion Prisms

The two fused silica Brewster angle prisms are fixed to their mount. The incident beam should strike the first prism about in the middle. Increase the laser power just enough so that all beams are visible.

Rotate the prism pair while watching the refracted beams some distance past the prisms. As the prisms are rotated the refracted beams will be seen to move in one direction and then reverse. The position at which the beams reverse is the place to stop. Rotate the prisms just slightly to either side of that maximum deflection point. In one direction it will be seen that the beams are more circular than in the other. Fasten the mount at the point near the maximum deflection where the beams are circular.

Put the mounted flint prism into its holder and adjust it in the same manner as the Brewster angle prisms.

2.6 Polarization Rotator

The polarization rotator should be set for maximum transmission through the dispersion prisms. A laser power meter will provide the greatest precision in this adjustment.

2.7 Inside Steering Mirrors

The three steering mirrors should be placed so that the incident beams strike them in the middle. The three mirrors direct the beams along a path of sufficient length to allow adequate separation of the beams. The mean beam height above the baseplate should be 4.25 inches. Some adjustment of the outside steering mirrors may be required. If the outside steering mirrors are adjusted, the other optical components should be checked and readjusted as necessary. A transparent ruler placed in the beam path will show the undiffracted beams below 4.25 inches and the diffracted beams an equal distance above 4.25 inches. The third mirror should be adjusted to send the beams onto the middle of the separation prisms. Ensure that all beams enter and exit cleanly without striking any edges.

2.8 Iris Diaphragms

In normal operation, the iris diaphragm should be wide open and set close to the third steering mirror. The two iris diaphragms define the position of the incident laser beam. After the positions of all components ahead of the second iris are set, reduce laser power to a minimum and switch off the Bragg cell driver. The only remaining beam will normally be the undiffracted blue beam. Now close down the first iris, center it over the single beam, and fasten it down. Open the first diaphragm and close down the second one. Position the second iris so it is centered on the single beam and fasten it in position. Open the iris, switch on all beams and ensure that all beams travel through the open iris. The two iris diaphragms can now be used to pre-position the system if alignment is lost.

2.9 Separation Prisms

Two large right angle prisms are used to further separate the diffracted and undiffracted (Bragged and un-Bragged) beams and direct them to the final individual prisms. The junction of the two prisms should be set to 4.25 inches up from the baseplate. Ensure that all beams travel through the large prisms cleanly. This is a good place to clip any unwanted short wavelength beams.

2.10 Final Steering Prisms

Each beam is picked off by a small right angle prism and directed to the fiber launching optics. The lower beams are 3.5 inches and the upper 5.5 inches above the baseplate. It is important that the final prisms be positioned so that the beams are directed squarely into the launching optics. Each beam should be orthogonal to the front plate which holds the launching optics. Transparent plastic rulers may be used to make this setting.

3.0 FIBER OPTIC LINK

3.1 Laser to Fiber Coupler

The laser to fiber couplers are mounted on the outside of the front panel by 1"-32 mounting threads. The function of the couplers is to launch the laser beams into the fibers efficiently. Each coupler will focus its respective laser beam down to a small waist and maneuver the single mode fiber to the image plane of the lens system. The coupler (Fig. 4) is comprised of two baseplates, each with an axial bore, with an O-ring sandwiched between. Lateral (radial) movement across the beam is accomplished by adjustment of the three small socket-head screws which compress the outer baseplate against an O-ring. The clearance holes for the three socket head screws in the outer plate are slightly oversized to allow some lateral movement when the screws are loose. Final precision adjustment and stability is provided by another three screws which push against the inner baseplate in opposition to the compressing screws. Z axis adjustment along the beam waist is provided by a fine threaded adjustment, which is locked down with a set screw.

3.2 Laser to Fiber Coupler Alignment

The procedure described applies to each beam and coupler.

WARNING

KEEP LASER POWER LOW UNTIL ALL COUPLERS ARE ALIGNED TO PREVENT BURNING FIBER CLADDING.

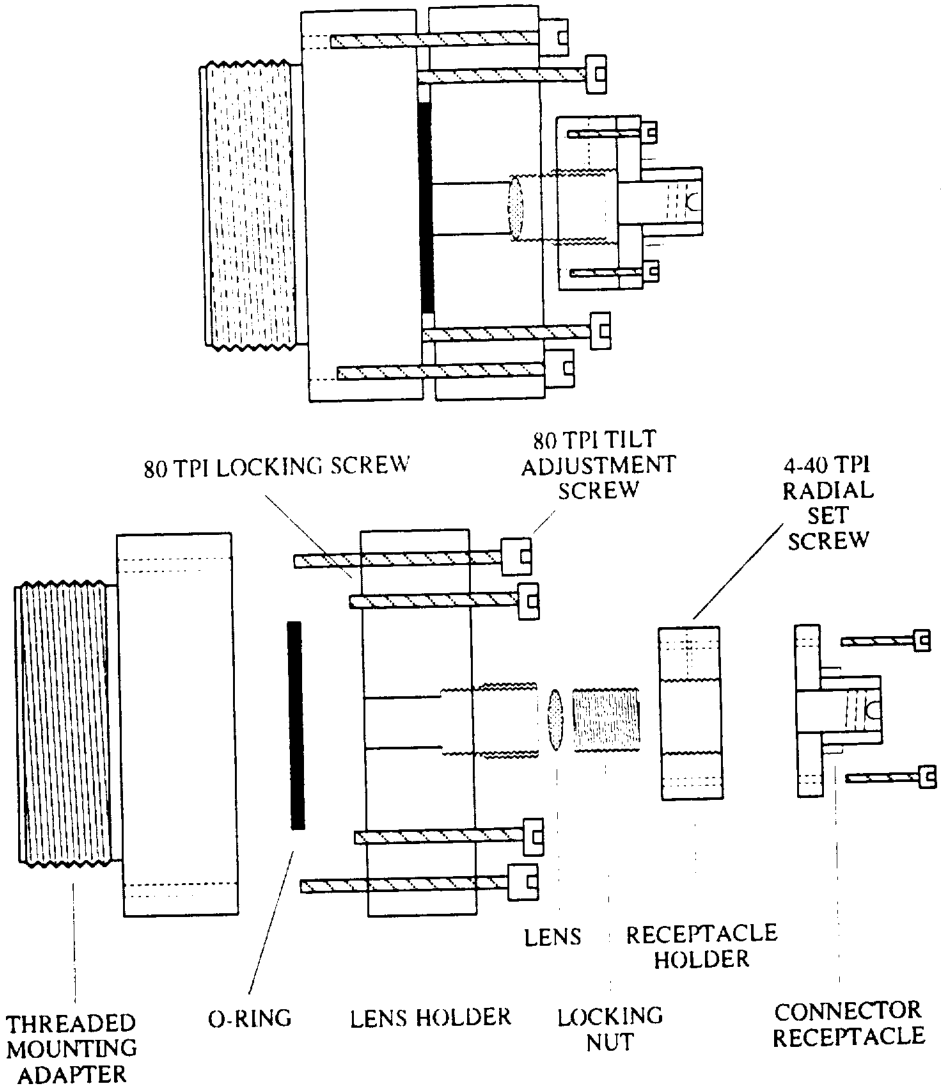


Figure 4 High Power Laser to Fiber Coupler

Operating the laser at low power, first center the beam in the front plate hole using a transparent plastic ruler or template, then screw in the coupler. Project the expanding laser beam onto a flat black screen one or two feet away. Slightly loosen the screws which hold the inner and outer coupler plates together to allow lateral movement. Displace the coupler disc by hand while observing the projected beam on the opaque screen. Tighten the screws in the position where the beam intensity is centered and symmetric. This aligns the lens axis with the beam, increasing coupling efficiency.

Insert the 50 micron multi-mode fiber into the coupler and set the fiber output in a position to project light onto an opaque screen (Fig. 5). Identify the three screws which pull the coupler plates together. The tilt mechanism will displace the fiber core laterally relative to the image at the focal plane. As the focused image nears the center of the core, lower numerical aperture (N.A.) modes will be excited and more light will be concentrated into the center of the output spot (Fig. 5b). This will occur only under launching conditions where the N.A. of the focused rays is

smaller than the N.A. of the fiber. Using the small ball driver, adjust capscrew 1 while observing the output. The distribution of light in the output should change. Try to concentrate most of the light into the center of the spot by rotating the capscrew. Adjust capscrew 2 and continue to concentrate more and more light into the modes closest to the center of the spot. Repeat with capscrew 3. Sequentially adjust each screw for the maximum light coupling while steadily pulling the coupler tighter.

When the coupler is quite tight and adjusted for maximum coupling, replace the multi-mode fiber with a single-mode fiber and optimize for maximum coupling efficiency. An optical power meter should be used for the final adjustments. At the focal plane, where the fiber is located, there are multiple maxima due to diffraction (Airy discs). If the maximum light coupled is very low (<10% of input) it might be that a side order maxima is positioned on the fiber core. While watching the power meter, tilt each of the screws sufficiently to verify that the most powerful maximum is being coupled into the fiber.

If coupling efficiency is still low, the Z-axis may need some adjustment. The easiest way to verify that the Z-axis does need adjustment is to loosen the FC type fiber optic connector (FC connector) a turn and pull the fiber back from the focal plane. Then slowly tighten the FC

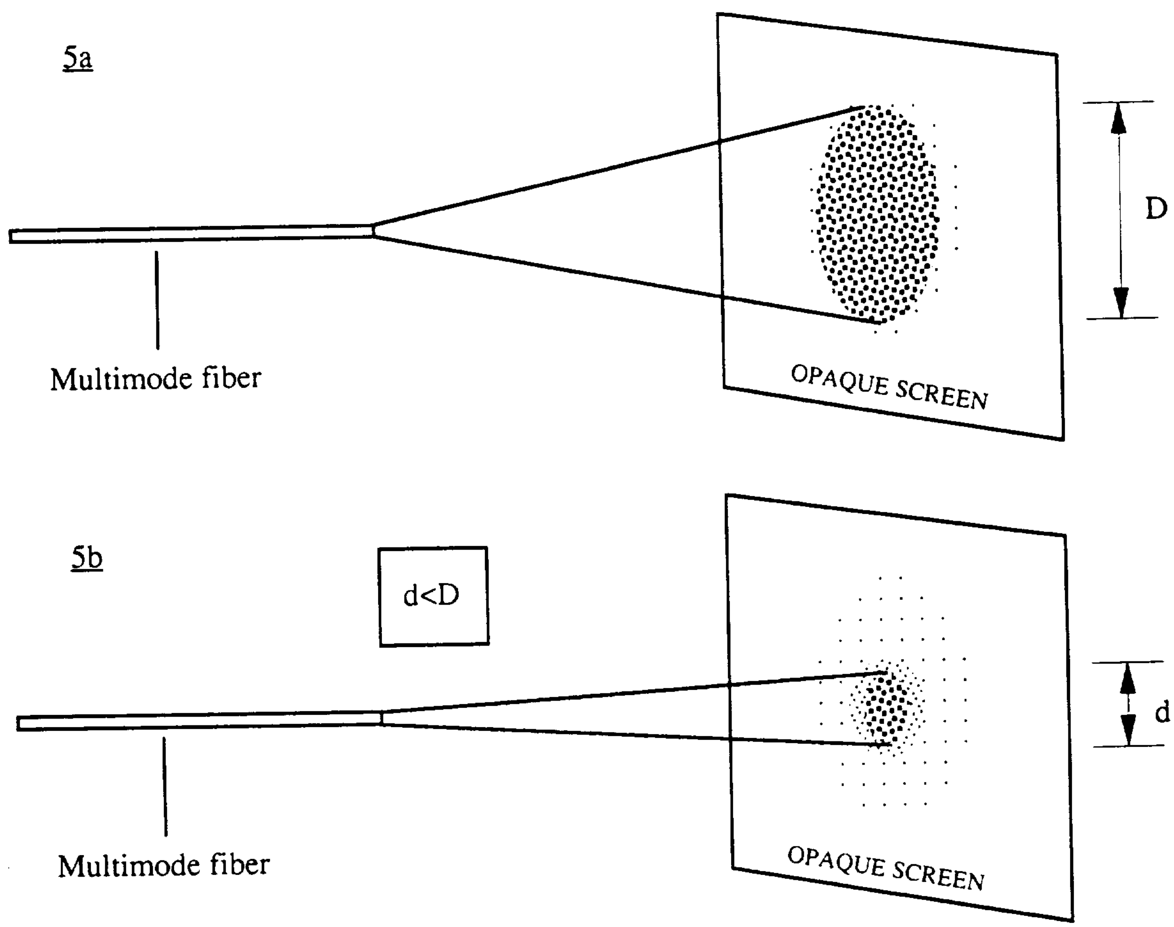


Figure 5 Multi-mode Fiber Pre-adjustment.

connector while watching the power meter to identify the position where the maximum coupling efficiency occurs. The aim is to peak coupling at the point when the FC connector is tight. If Z axis adjustment is necessary, alignment may be maintained by making exact 360 degree adjustments; that is, loosen or tighten the Z axis by exactly one turn. If alignment is lost or if more than a minor adjustment is necessary, the multi-mode fiber should be installed initially. If significant adjustment is necessary, the procedure described in the section on preliminary Z-axis adjustment should be followed.

3.3 Preliminary Z-Axis Adjustment

The Z-axis is pre-adjusted and this procedure should not normally be required. If adjustment becomes necessary, the procedure is to adjust the coupler as you would a collimator. Couple light into a fiber. Install the launching coupler on the output end of the fiber. Loosen the radial set screws which secure the Z-axis ferrule. Project the beam onto an opaque screen and adjust the Z-axis ferrule until you get a minimum diameter collimated beam at some distance. The lens is now positioned so that the fiber is at its focal plane. The coupler is now set up to launch a collimated input beam when used as a laser to fiber coupler.

3.4 Polarization Axis Adjustment

The single mode fibers provided with this system are highly birefringent, polarization-maintaining fibers, generally with two perpendicular principal axes. By maintaining polarization in the fibers we are able to match the polarization of the beam pairs in the measurement volume. Also, the output of properly aligned polarization-maintaining fiber will not fluctuate when the fibers are moved or manipulated. Polarization is maintained only when the polarization axis of the light is matched with that of the fiber. Improper alignment will cause the output polarization state of the fiber to oscillate between elliptical and linear polarization states. The polarization of the light can be rotated using a half wave plate placed ahead of the launching optics to match the orientation of the fiber or the fiber can be rotated to match the polarization orientation of the light.

Polarization alignment of fibers is measured by determining the extinction ratio of the output. First align the coupler for best coupling efficiency. Measure the fiber output through a polarizer with a light powermeter. Rotate the polarizer until maximum light transmission through the polarizer is achieved. Record this value. Rotate the polarizer until the minimum output is achieved and record the powermeter reading. Calculate the difference (extinction ratio) between the maximum and minimum readings in dB. Then rotate the knurled section of the fiber connector (Fig. 6) very slightly and repeat the procedure. Continue to rotate the fiber connector until the extinction ratio is maximum. Extinction ratios of 20 to 35 dB should be achieved. As mentioned above, a half wave rotator placed ahead of the launching optics in a rotary mount can easily set the light polarization to match the principal axis of the fiber. Placed in position temporarily, a half wave rotator can be used to help determine if the best polarization matching has been achieved or approximately how much adjustment is required. When the best extinction ratio has been

achieved, press or bend the fiber slightly, which may result in a small change in the power output after the polarizer. If the change is less than a few dBs, the polarization axes are aligned. Ideally, there should be no change. If the change is more than a few dBs, then rotate the fiber connector slightly until the required extinction ratio is achieved.

When the fiber is rotated, if an increase in insertion loss is noticed, it is due to fiber core / cladding concentricity problems. In lens style couplers, this could be compensated for by adjusting the angle between the incoming collimated beam and the receiver lens. Adjustment as described above in the previous section should be performed.

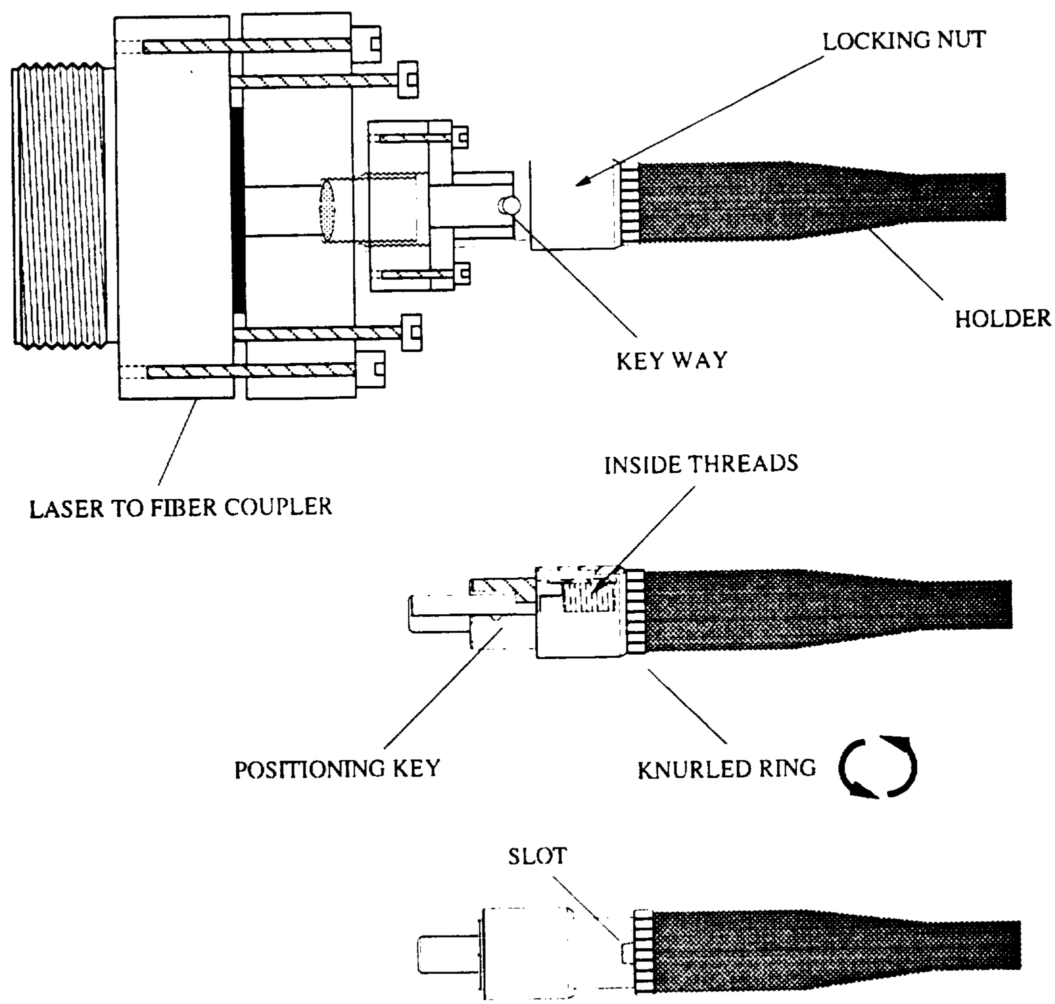


Figure 6 Polarization Preserving Connector.

Once the polarization axes have been properly set, the connectors may be glued to fix them in position. Loctite 290® for preassembled fasteners, Duco® cement or instant glue may be used,

depending on the degree of permanence desired. The glue should be put in the slot as shown at the bottom of Fig. 6.

3.5 Plenum Feedthrough for Optical Fibers

For additional protection, the armored fiber optic cable is contained within a conduit. Figure 7 shows how the fiber and conduit was plumbed through the plenum portholes. Inside the plenum, shop air was blown through the conduit to cool the fiber.

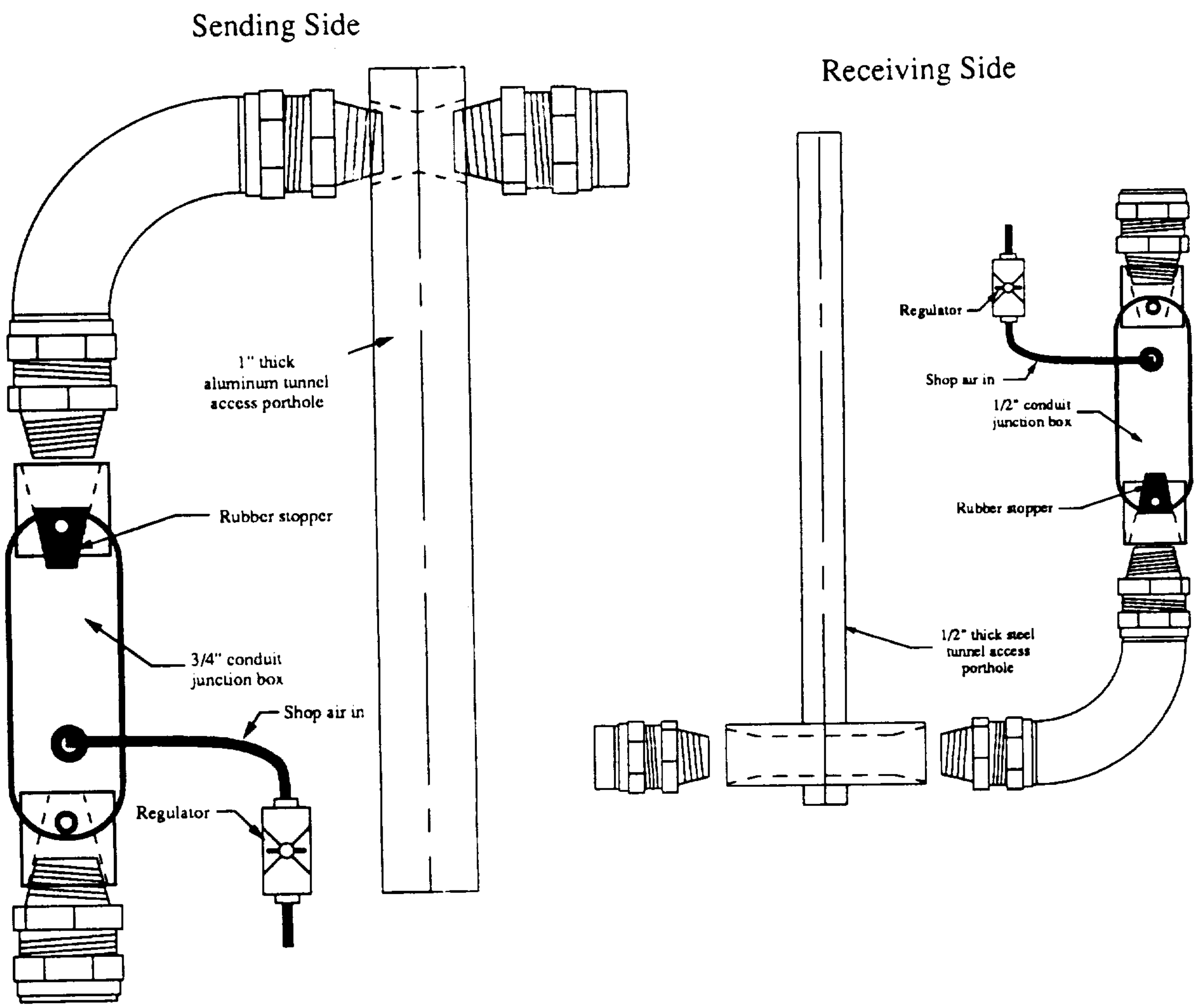


Figure 7 Plenum Feedthrough for Optical Fibers.

4.0 FIBER COLLIMATORS AND TRANSMITTING OPTICS

A fiber collimator is very similar in design to a laser to fiber coupler. Collimator length is proportional to the output beam diameter desired. The output beam is collimated by releasing the set screw(s) and adjusting the Z-axis, which is the distance from the fiber output to the collimating lens. This is best done before the collimator is attached to the mounting plate. Direct the beam some distance away and adjust the Z-axis until the best collimation is achieved.

With the output of each collimator set, they are attached to the mounting plate with three screws and an o-ring. To set the four beams mutually parallel, set the mounting plate in its mounting ring in a stand on the optics table. Set up a 60 mm template at the same height several feet away. Adjust each collimator to steer its beam onto the appropriate spot. The three mounting screws on each collimator should be quite snug. When adjustment is complete, tighten the three locking screws while checking that alignment is maintained. Repeat the procedure for the other mounting plate.

Fasten the collimator plate into the 100 mm I.D. mounting ring, then to the rail. The rhomboid beam separation reducer should be fastened to the rail ahead of the collimator plate. The beam separation used for this application was .3125 inches. Ensure clean passage of the laser beams through the rhomboids. Lastly, the focusing lens should be attached to the rail.

WARNING

WEAR LASER SAFETY EYEWEAR.

BEWARE OF SPECULAR REFLECTIONS.

Using a Polaroid filter, check that polarization of each beam pair is matched. Output polarization is set by rotating the knurled ring on the FC connector (Fig. 6). The fiber axis and beam polarization at the fiber input should have been matched already, as described above in the section on polarization axis adjustment.

With a clear plastic ruler, check that all beams cross at the focal point of the converging lens. Be sure to use appropriate eye safety precautions. Place a microscope objective or eyepiece in the beams to project the crossover point. Move the eyepiece axially along the beams. Each beam should be waisting at the focal volume. Use the Z-axis adjustment of each collimator to set the beam waist. Notice that the tightness of the FC connector at the collimator affects the collimation and thus the beam waist somewhat.

Each beam pair should be crossing fully without shearing and all pairs should cross together. For fine adjustment use the three screws on each collimator which press the plates apart.

5.0 COLLECTING OPTICS

The scattered light collection lens should be positioned for optimum forward scatter collection. A thin scattering center such as a piece of tape should be placed near the center of the proposed scan and the collecting optics driven and adjusted to focus the collected light into the multi-mode fiber. Direct the output of the multi-mode fiber onto a flat black surface. When the fiber is at the focus of the collecting optics, the center of the projected fiber output is illuminated. If the fiber is not located at the focus, there will be a bright ring around the outer edge of the illuminated area.

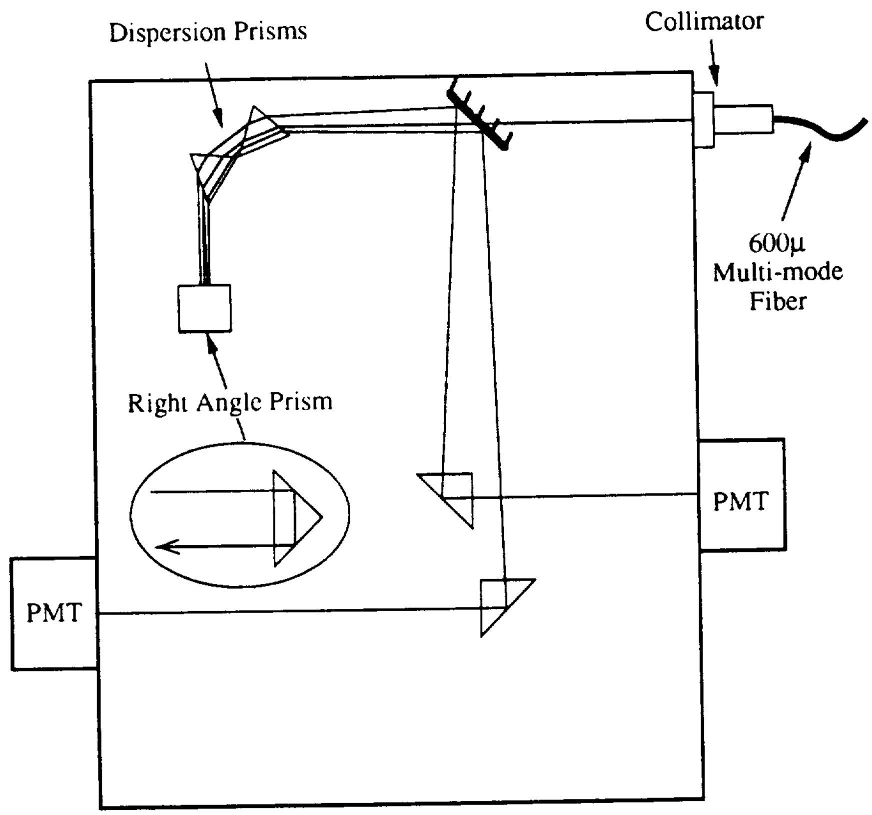


Figure 8 Collected Light Color Separation and Signal Detection System.

6.0 COLLECTED LIGHT COLOR SEPARATION SYSTEM

Figure 8 illustrates the layout for the collected light color separation and signal detection system. The multi-mode fiber collimator is adjusted in a similar fashion to the single mode unit. To set beam collimation, the set screw on the shaft is loosened and the distance from fiber end to collimating lens is varied while viewing the beam size at some distance. The collimator is then

screwed into the color separation box using the threaded 1"-32 hole. The collimated beam is directed through a pair of dense flint dispersion prisms to a right angle prism, which reverses the beam to travel through the dispersion prisms a second time at a slightly lower level. Another right angle prism directs the diverging colors through sets of light baffles to the final steering prisms which send each color to its photo-multiplier tube.

The dispersion prisms are fixed to their mount. During initial alignment, the dispersion prism pair, together with the direction reversing right angle prism, were positioned to provide the greatest dispersion with unclipped beams. If a component is knocked out of alignment, the best course is usually to leave the other components undisturbed and replace and adjust that component until proper orientation is again achieved.

CHAPTER 2

LASER VELOCIMETER DATA ACQUISITION SYSTEM.

CHAPTER 2

Laser Velocimeter Data Acquisition System.

TABLE OF CONTENTS

LIST OF FIGURES

1.0 INTRODUCTION

The NASA Ames Research Center 3.5 Foot Hypersonic Wind Tunnel Laser Doppler Velocimeter System provides the capability to acquire and process simultaneous analog data and two-component Laser Doppler Velocimeter (LDV) data. The system consists of the following five sub-systems:

1. LDV Signal Conditioning Instrumentation.
2. LDV Counter Signal Processor Instrumentation.
3. Laser Velocimeter Data Acquisition System (LVDAS).
4. Data Acquisition, Data Reduction, and Data Presentation Computer System.
5. Traverse Control System (TCS8).

This document will discuss the theory of operation of the LVDAS and the LDV Counter Signal Processors as well as provide sources of documentation drawings for these instruments. The manner in which they are connected to and interact with each of the other optical and electronic sub-systems listed above will also be included. Figure 1 shows the configuration setup of the Laser Doppler Velocimeter system. This shows how the LVDAS fits into the complete system.

1.1 LDV Signal Conditioning Instrumentation.

Figure 2 shows the signal conditioning that is applied to the two-component Laser Doppler Velocimeter signals, the tunnel static temperature signal, and other optional analog voltage signals. The tunnel static temperature voltage output is fed directly to the first analog input channel of the LVDAS. Other optional analog voltage outputs can also be fed directly to one of the analog inputs of the LVDAS.

The LDV signal conditioning instrumentation is composed of the following elements:

1. RF Amplifier.
2. Frequency Filter.
3. Macrodyne LDV Counter Processors.

The voltage outputs of each RF amplifier are fed directly to the inputs of the Macrodyne LDV Counter Processors. The 16bit digital outputs of the two Macrodyne LDV Counter Processors are connected directly to the digital inputs of the LVDAS. The LVDAS is described in Section 2 while the Macrodyne LDV Counter Processors are described in Sections 3 and 4.

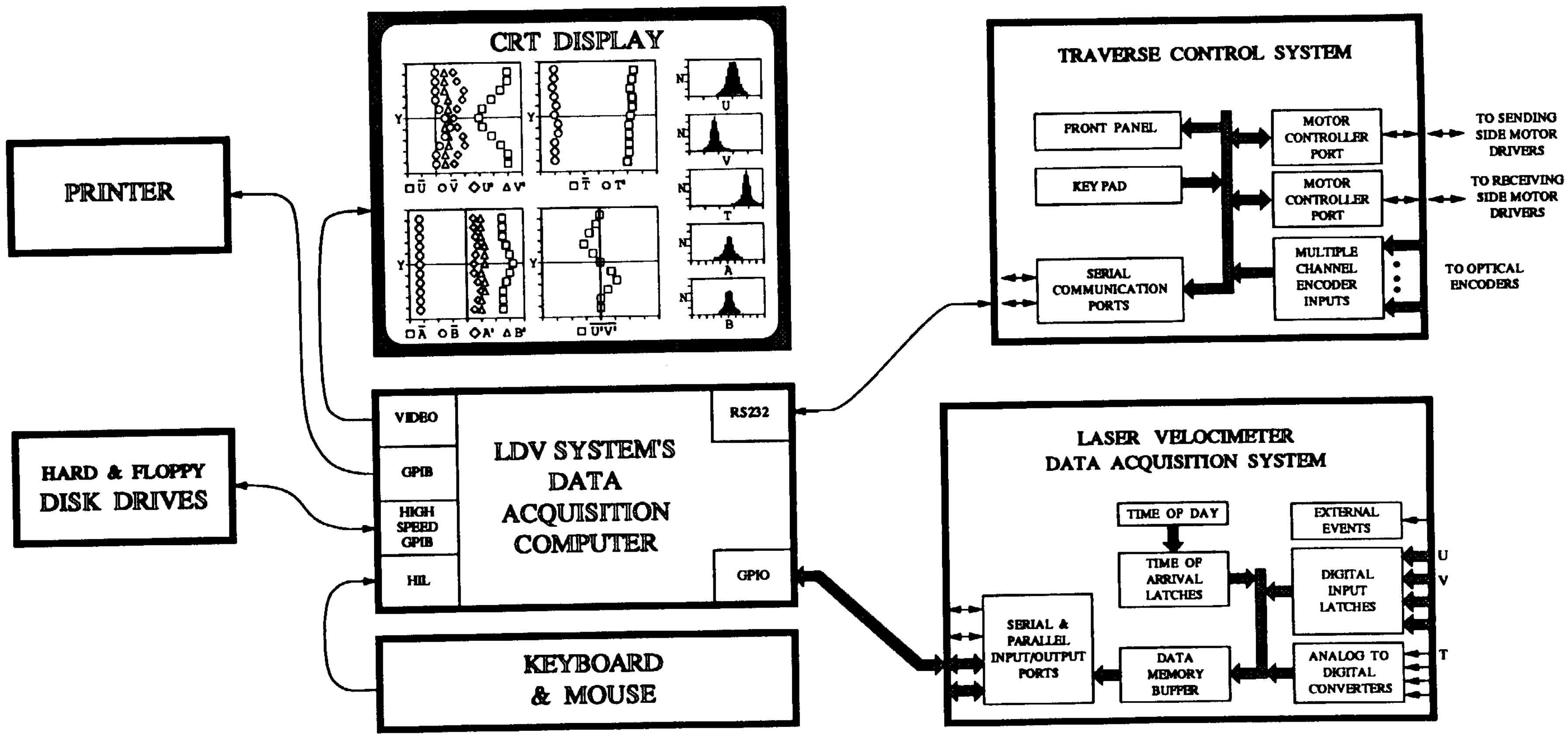


Figure 1. Laser Velocimeter System Configuration.

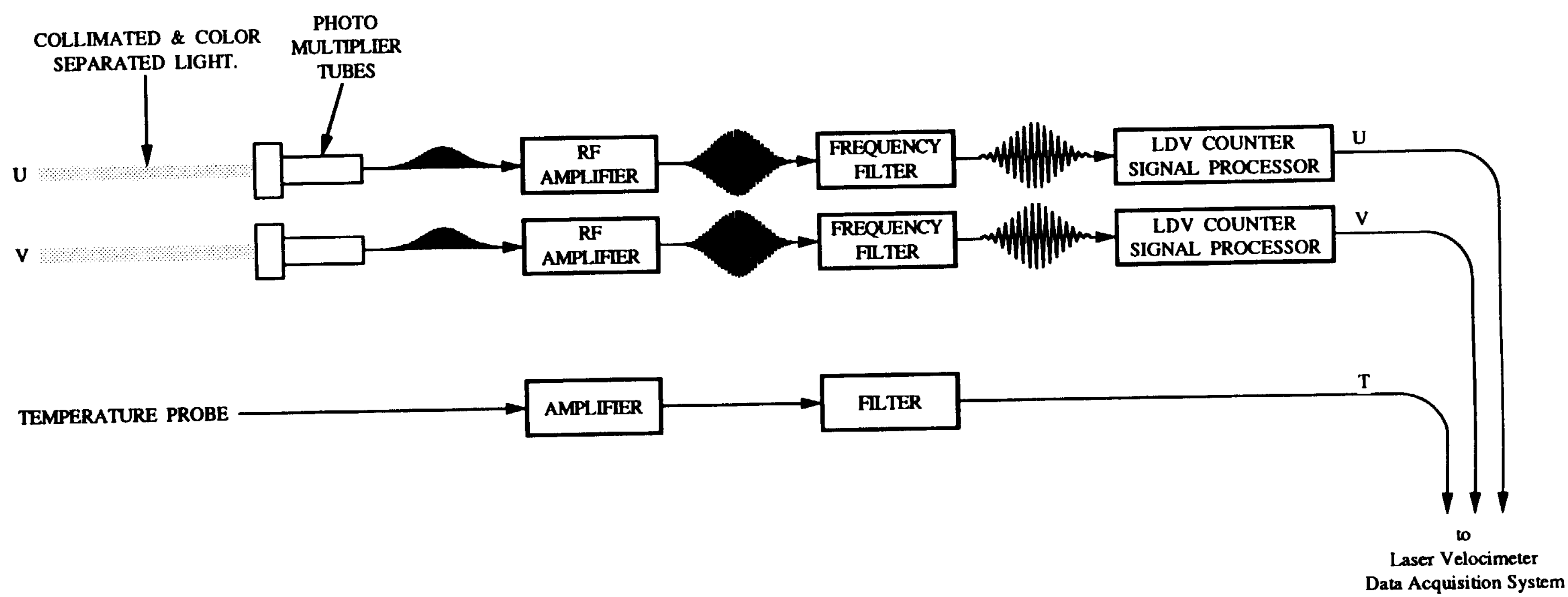


Figure 2. Two-Component LDV Signal Conditioning.

2.0 LASER VELOCIMETER DATA ACQUISITION SYSTEM.

New applications in laser velocimetry have brought about the need for a more advanced laser velocimeter data acquisition system. These new applications require high data rates that are not hindered by on-line time dependent data sorting and real time graphic data presentation. The new Laser Velocimeter Data Acquisition System (LVDAS) was designed specifically to meet these advanced requirements.

The Laser Velocimeter Data Acquisition System (LVDAS) provides the capability to acquire, process, and present real time digital data and analog data. The digital, for LDV systems, is typically the output of LDV signal processors. The analog data, for hypersonic wind tunnels, might include the raw signals containing tunnel temperature and/or pressure. The output of a filter amplifier whose input comes from flow sensors, such as a hot wire, might also be acquired with the LDV data. Additional analog data might originate from such sources as temperature probes, position sensors, etc. A functional schematic diagram of the LVDAS is shown in Figure 3. The LVDAS acquires simultaneous digital data, analog data, and time information data. The data are sampled, multiplexed, buffered, and then transferred to the facility's host computer for further data reduction, analysis, and presentation.

The digital data are sampled in a manner which ensures that the required coincidence time criterion is met. This is achieved by comparing 32bit 10MHz time of arrival counters for each of the digital channels. If data arrives on all of the selected digital channels within the coincidence time, then the analog channels are immediately sampled and converted. Otherwise, the digital data are rejected and the process is repeated. The 16 bit word parallel input ports are provided to accept the digital output of LDV counter processors and/or other instrumentation. High data acquisition rates are achieved by providing a separate latched input for each laser velocimeter digital input and a separate converter for each temperature, pressure, hot wire sensor or other analog input. The system will allow for data acquisition rates of approximately 100,000 samples per second simultaneously on each of the laser velocimeter and analog inputs.

A 32 bit time of day (TOD) 10MHz counter is used to tag arrival times to acquired digital LDV data as they become available on each of digital inputs. When a data valid "sync" pulse is sensed for a particular channel, the LVDAS latches the current TOD into a 32 bit time of arrival register (TOA). A separate TOA register is available for each digital input, so that particle arrival times of measured velocity information for U,V, and W can be monitored for coincidence. The latched times of arrivals have a resolution of 100 ns and maximum time of over 7 minutes.

The coincidence control logic allows for up to 3 channel coincidence. The coincidence time can be adjustable to any resolution or duration within the capability of the time of arrival registers. The coincidence time is adjustable from 100 ns to 1 s. In addition to the laser velocimeter inputs, three additional data words are generated internally. They are the inter-arrival time, the coincidence time, and status words. The inter-arrival and coincidence time is provided by a clock whose resolution is 100 ns and the maximum elapsed time is over 7 minutes. The status word contains information about coincidence which indicates whether or not valid data have been acquired.

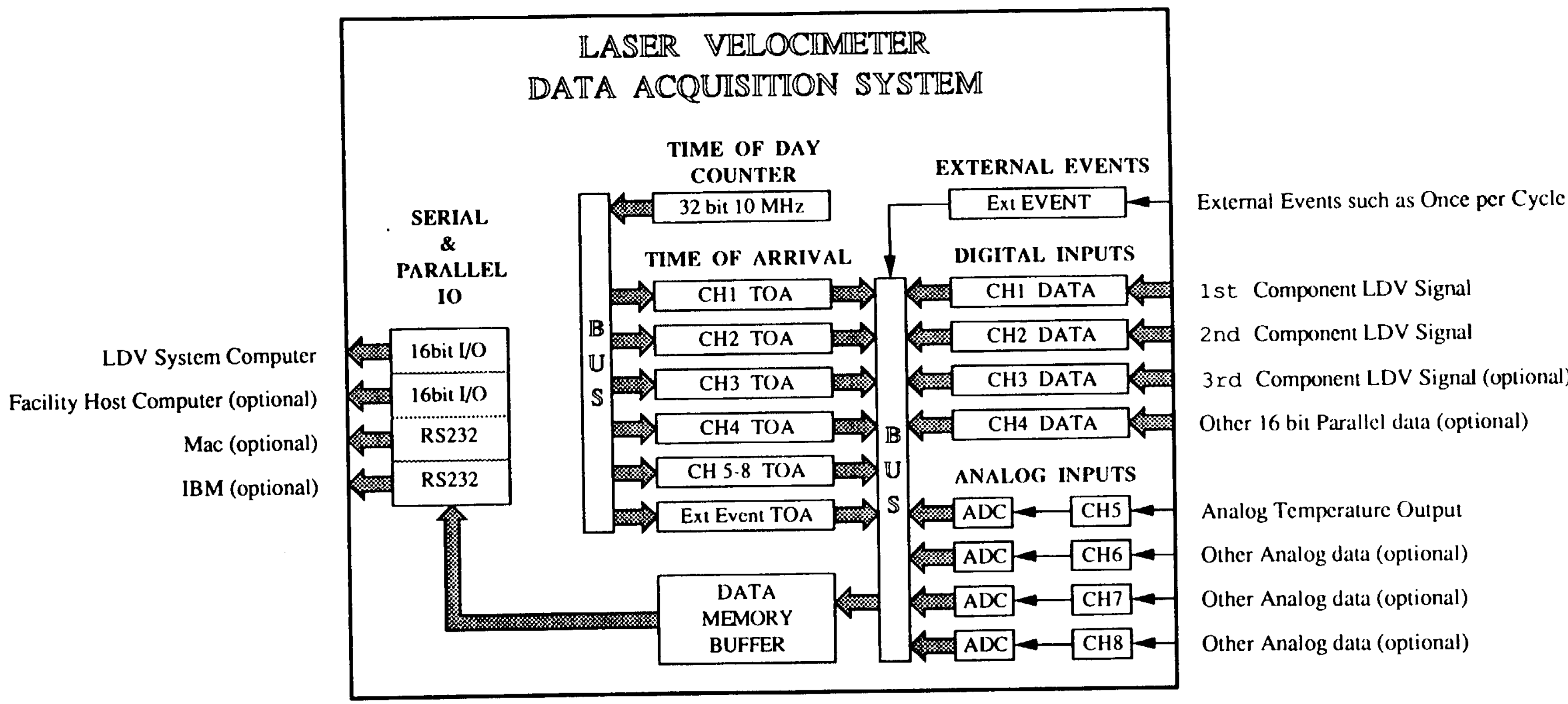


Figure 3. Laser Velocimeter Data Acquisition System.

When coincident criteria are met, the analog inputs can be sampled and converted to provide concurrent data with the digital data. A single time of arrival is latched for all each of the analog to digital inputs, since they are all sampled and converted simultaneously. Additionally, a time of arrival is latched for external events, if they occur. These might be derived from such sources as oscillating models or model surfaces, rotating helicopter blades, rotating engine fans, or flow sensors.

All of the acquired digital velocity data with corresponding time of arrival data can be processed and stored even if coincidence is not required. However, if coincident data are required, then the arrival time of the various channels can be conditionally accepted if they all occur within a finite window of time. These coincident events can then be assigned inter-arrival times, which represent elapsed time since the previous event.

During data acquisition, it is important that the user obtain some visual feedback about the data being acquired. This is necessary so that the user can make informed decisions about both the quality and quantity of data received. The user is either reassured about the quality of the data or can make alterations and improvements in technique while "on line". To help achieve this, the instantaneous velocities are used to generate real time histograms from which probability density distributions are determined for all velocity components.

Additionally, the laser velocimeter data acquisition system has the capability of reducing the raw laser velocimeter data. Each laser velocimeter output contains the information required to calculate the instantaneous velocities. From the instantaneous velocity determinations, the average velocities, turbulence levels, and the turbulence cross correlations are all to be calculated.

All digital Macrodyne data, optional digital data, analog to digital data, and time of arrival data can be sent by the LVDAS to other computers via two serial and two parallel input/output ports. One parallel port will be used for the LDV system's data acquisition computer while the other can be used by the facility host computer. The serial ports can be used by PC type computers such as IBMs or MACs.

2.1 Analog Data Description.

The analog inputs are provided to accept differential voltages from such sources as hot wires, temperature probes, pressure probes, and other such sensors and/or instrumentation. The inputs are differential inputs and accept ±5 volts.

The inputs are sampled and converted at a rate 200KHz with 16 bit resolution. The converted raw data (16 bit word) are encoded into signed two's complement binary format. Examples of the raw data to voltage conversion is shown on the next page.

msb BINARY WORD lsb	INTEGER	VOLTAGE
0111 1111 1111 1111	32767	+4.99985
0111 1111 1111 1110	32766	+4.99969
⇓ ⇓ ⇓ ⇓	⇓	⇓
0000 0000 0000 0001	1	+0.00015
0000 0000 0000 0000	0	0.00000
1111 1111 1111 1111	-1	-0.00015
⇓ ⇓ ⇓ ⇓	⇓	⇓
1000 0000 0000 0001	-32767	-4.99985
1000 0000 0000 0000	-32768	-5.00000

One Bit Resolution: 0.00015

The signed binary two's complement integer word can be converted to a real precision floating point voltage using the following equation.

$$A_V = \frac{5A_R}{2^{15}} \quad \text{volts}$$

Where A_R is the analog raw voltage and A_V is the converted analog voltage.

2.2 Digital Data Description.

Three digital inputs are provided to accept 16bit digital data from such sources as LDV counter signal processors and/or instrumentation. The format for the Macrodyne Counter Signal Processors, which are being used with this system, will depend on whether or not the old or new model Macrodynes are used. The old models provide 16 bits of frequency information on a DB type 25 socket connector. The new models provide 18 bits of frequency information on a DB type 37 socket connector. The data formats and cable schematics for the old models are included in Section 3. The data formats and cable schematics for the new models are included in Section 4. (Note: The new model Macrodynes were delivered with the two-component LDV system. Therefore, the description of the "New Model Macrodynes" in section 4.0 and the cable schematic shown in Figure 6 would apply to this system.)

3.0 <u>OLD MODEL MACRODYNES.</u>

This chapter describes the data format of the old model Macrodyne's digital output port. Also described are the equations necessary to convert the raw data into frequency which represents the rate of fringe crossings of the Doppler burst. Additionally, a detailed description and schematic drawing is provided for the Macrodyne to LVDAS interface cable (see Figure 5). Figure 7 is a timing diagram showing the handshaking sequence of the control lines. Figure 7 also shows the timing sequence of the data lines. This indicates when the data become valid and then later latched.

3.1 Data Format.

The old model Macrodyne LDV counter signal processors provide the digital frequency output in the following 16 bit format:

```
MSB                                                          LSB
15    14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
TIME 5/8  X3  X2  X1  X0  D9  D8  D7  D6  D5  D4  D3  D2  D1  D0
```

The mantissa (D9..D0) is contained within the lower 10 bits of the 16 bit word while the exponent (X3..X0) is contained within bits 10 through 13. The number of fringes measured is defined as 8 if 5/8=1 or 16 if 5/8=0. The time bit tells whether the mantissa and exponent represent a period (TIME=0) or a velocity (TIME=1). The period of the Doppler frequency is measured by the number of clock pulses (C=500MHz or 1000MHz) that are required to sense 8 or 16 fringe crossings.

The one bit 5/8 bit is set by the front panel 5/8 - 10/16 switch. It specifies whether 8 or 16 fringes are to be measured by the counter processor. The one bit TIME bit is set by the front panel Time/Velocity switch. It specifies one of two encoding schemes used by the Macrodynes to represent the frequency data.

The logic level for each of the old model Macrodynes varies with different units. Each unit may have a mixture of positive true or negative true logic for the TIME, 5/8, X3..X0, and D9..D0 data pins on the digital output port. Typically, when one orders multiple units, they all come configured with the same logic. But, units ordered at a later date may have a different mixture of positive true and negative true logic on the digital output port data pins.

3.2 Frequency Mode.

If TIME=0 then the data represent the time that had elapsed while a specified number of fringe crossings were observed by the counter processor. The elapsed time is encoded into a 10 bit mantissa (D9..D0) and 4 bit exponent (X3..X0). Both the mantissa and the exponent are in unsigned binary format.

3.3 Velocity Mode.

If TIME=1 then the data represents the frequency of the observed doppler burst. The frequency information is encoded into a 14 bit concatenation of the 4 bit exponent (X3..X0) and the 10 bit mantissa (D9..D0). The resulting 14 bit frequency word (X3..X0 D9..D0) is in unsigned binary format.

3.4 Macrodyne Front Panel Digital Output Pinouts.

The pinout assignment for the old model Macrodynes is shown in the first column of Figure 4.

3.5 Interface Cable Schematic and Handshake Timing Diagram.

Figure 5 shows a detailed schematic drawing for the Macrodyne to LVDAS 16bit parallel data interface cable. Figure 7 is a timing diagram of the handshake processes that happen each time data are transferred from the Macrodyne to the LVDAS.

3.6 Data Reduction.

The following sections describe how to convert the raw data into useful period or frequency data. The raw data are encoded into the 5/8 fringe count bit, 4 bit X3..X0 period exponent, and 10 bit D9..D0 period mantissa.

3.7 Period Calculation

The time T for the selected number of fringes can be calculated using the following equation. (Note: T is the time for the entire measured burst of 8 or 16 fringes.)

$$T = M2^{(E-2)} \qquad \text{ns}$$

Where M is the mantissa bits D9 through D0 and E is the exponent bits X3 through X0. To determine the doppler period for 8 fringes (T_8) and for 16 fringes (T_{16}) the following equations would apply. (Note: Both T_8 and T_{16} are the average time for only one fringe of the entire measured burst of 8 or 16 fringes.)

$$T_8 = \frac{M2^{(E-2)}}{2^3 10^9} \quad \text{s}$$

$$T_{16} = \frac{M2^{(E-2)}}{2^4 10^9} \quad \text{s}$$

3.8 Frequency Calculation

The doppler frequency can be calculated using one of the following equations depending on whether 8 of 16 fringes were measured.

$$F_8 = \frac{1}{T_8} = \frac{2^3 10^9}{M2^{(E-2)}} \quad \text{Hz}$$

$$F_{16} = \frac{1}{T_{16}} = \frac{2^4 10^9}{M2^{(E-2)}} \quad \text{Hz}$$

3.9 Velocity Mode Frequency Calculation

The Macrodyne manuals provide no information as to the conversion of the raw velocity mode data into useful frequencies of velocities. Therefore, no equations are provided here for raw data to frequency conversion. This data format is not used at the present time.

4.0 NEW MODEL MACRODYNES

This chapter describes the data format of the new model Macrodyne's digital output port. Also described are the equations necessary to convert the raw data into frequency which represents the rate of fringe crossings of the Doppler burst. Additionally, a detailed description and schematic drawing is provided for the Macrodyne to LVDAS interface cable (see Figure 6). Figure 7 is a timing diagram showing the handshaking sequence of the control lines. Figure 7 also shows the timing sequence of the data lines. This indicates when the data become valid and then later latched.

4.1 Data Format.

The new model Macrodyne LDV counter processors provide frequency information in a similar format to the old models. New model Macrodyne LDV counter signal processors provide the digital frequency output in the following 18 bit format:

MSB																	LSB
17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME	5/8	X3	X2	X1	X0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

The mantissa (D11..D0) is contained within the lower 12 bits of the 18 bit word while the exponent (X3..X0) is contained within bits 12 through 15. The number of fringes measured is defined as 8 if 5/8=1 or 16 if 5/8=0. The time bit tells whether the mantissa and exponent represent a period (TIME=0) or a velocity (TIME=1). The period of the Doppler frequency is measured by the number of clock pulses (C=500MHz or 1000MHz) that are required to sense 8 or 16 fringe crossings.

The major difference is that the addition of two more mantissa bits (D11 and D10) to provide increased dynamic range for a fixed exponent. The equations for determining T, T8, T16, F8, and F16 would be identical to those illustrated in the previous section. However, D0 and D1 are ignored, since the current LVDAS digital interface provides for 16 input lines. Therefore, the following 16 bit format is actually transmitted to the LVDAS.

MSB															LSB
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME	5/8	X3	X2	X1	X0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2

The one bit 5/8 bit is set by the front panel 5/8 - 10/16 switch. It specifies whether 8 or 16 fringes are to be measured by the counter processor. The one bit TIME bit is set by the front panel Time/Velocity switch. It specifies one of two encoding schemes used by the Macrodynes to represent the frequency data.

4.2 Frequency Mode.

If TIME=0 then the data represent the time that had elapsed while a specified number of fringe crossings were observed by the counter processor. The elapsed time is encoded into a 12 bit mantissa (D11..D0) and 4 bit exponent (X3..X0). Both the mantissa and the exponent are in unsigned binary format.

4.3 Velocity Mode.

If TIME=1 then the data represents the frequency of the observed doppler burst. The frequency information is encoded into a 16 bit concatenation of the 4 bit exponent (X3..X0) and the 12 bit mantissa (D11..D0). The resulting 16 bit frequency word (X3..X0 D11..D0) is in unsigned binary format.

4.4 Front Panel Digital Output Pinouts.

The pinout assignment for the old model Macrodynes is shown in the second column of Figure 4.

4.5 Interface Cable Schematic and Handshake Timing Diagram.

Figure 6 shows a detailed schematic drawing for the Macrodyne to LVDAS 16bit parallel data interface cable. Figure 7 is a timing diagram of the handshake processes that happen each time data are transferred from the Macrodyne to the LVDAS.

4.6 Date Reduction.

The following sections describe how to convert the raw data into useful period or frequency data. The raw data are encoded into the 5/8 fringe count bit, 4 bit X3..X0 period exponent, and 12 bit D11..D0 period mantissa.

With the deletion of the D1 and D0 the mantissa M would be represented by D11..D2 instead of D11..D0 and equations for T, T_8, T_{16}, F_8, and F_{16} would be modified as shown in Sections 4.7 and 4.8.

4.7 Period Calculation

$$T=M2^{(E-0)} \qquad \text{ns}$$

$$T_8 = \frac{M2^{(E-0)}}{2^3 10^9} \qquad \text{s}$$

$$T_{16} = \frac{M2^{(E-0)}}{2^4 10^9} \qquad \text{s}$$

4.8 **Frequency Calculation.**

$$F_8 = \frac{1}{T_8} = \frac{2^3 10^9}{M2^{(E-0)}} \qquad \text{Hz}$$

$$F_{16} = \frac{1}{T_{16}} = \frac{2^4 10^9}{M2^{(E-0)}} \qquad \text{Hz}$$

4.9 **Velocity Mode Frequency Calculation.**

The Macrodyne manuals provide no information as to the conversion of the raw velocity mode data into useful frequencies of velocities. Therefore, no equations are provided here for raw data to frequency conversion. This data format is not used at the present time.

	3002 (OLD) FRONT	3002 (NEW) FRONT	300X REAR	3003 FRONT
1	-D0	D00	-D0	-D0
2	-D2	D02	-D2	-D2
3	-D4	D04	-D4	-D4
4	-D6	D06	-D6	-D6
5	-D8	D08	-D8	-D8
6	-X0	D10	-X0	-X0
7	-X2	X0	-X2	-X2
8	-5/8	X2	-5/8	-5/8
9	SYNC	-HOLDOFF	-SYNC	-SYNC
10	-INHIBIT	SYNC	-SprInh	-INHIBIT
11	+5V	TIME	-RESET	-CC1
12	-TIME	NotUsed	10MHz	-CompAcc2
13	GROUND	NotUsed	-CC2	-CompAcc8
14	-D1	GROUND	-D1	-D1
15	-D3	GROUND	-D3	-D3
16	-D5	NotUsed	-D5	-D5
17	-D7	NotUsed	-D7	-D7
18	-D9	NotUsed	-D9	-D9
19	-X1	+5V	-X1	-X1
20	-X3	D01	-X3	-X3
21	GROUND	D03	GROUND	GROUND
22	GROUND	D05	GROUND	GROUND
23	GROUND	D07	GROUND	GROUND
24	GROUND	D09	GROUND	-CompAcc1
25	GROUND	D11	GROUND	-CompAcc4
26		X1		
27		X3		
28		-ExtRes		
29		AnologOut		
30		-EIGHT		
31		NotUsed		
32		GROUND		
33		GROUND		
34		GROUND		
35		NotUsed		
36		NotUsed		
37		NotUsed		

Figure 4. Macrodyne Digital Output Port Pinouts.

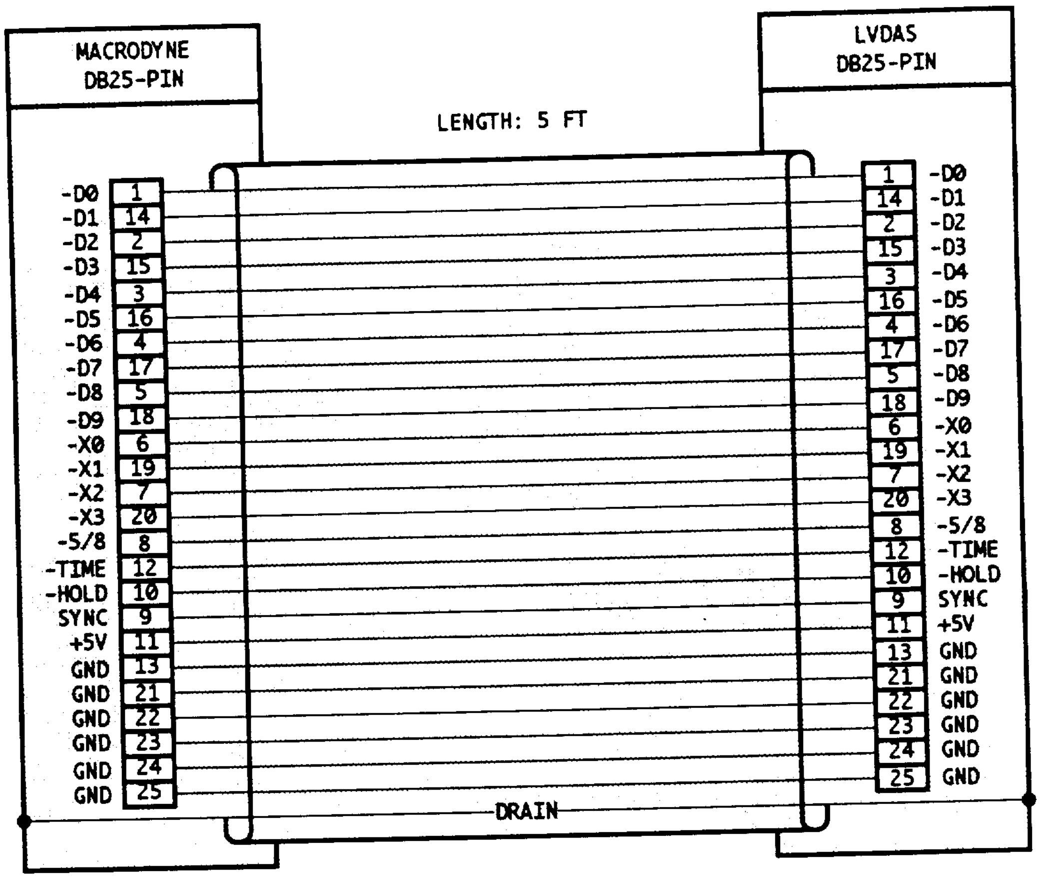


Figure 5. Old Macrodyne to LVDAS Interface Cable Schematic Drawing.

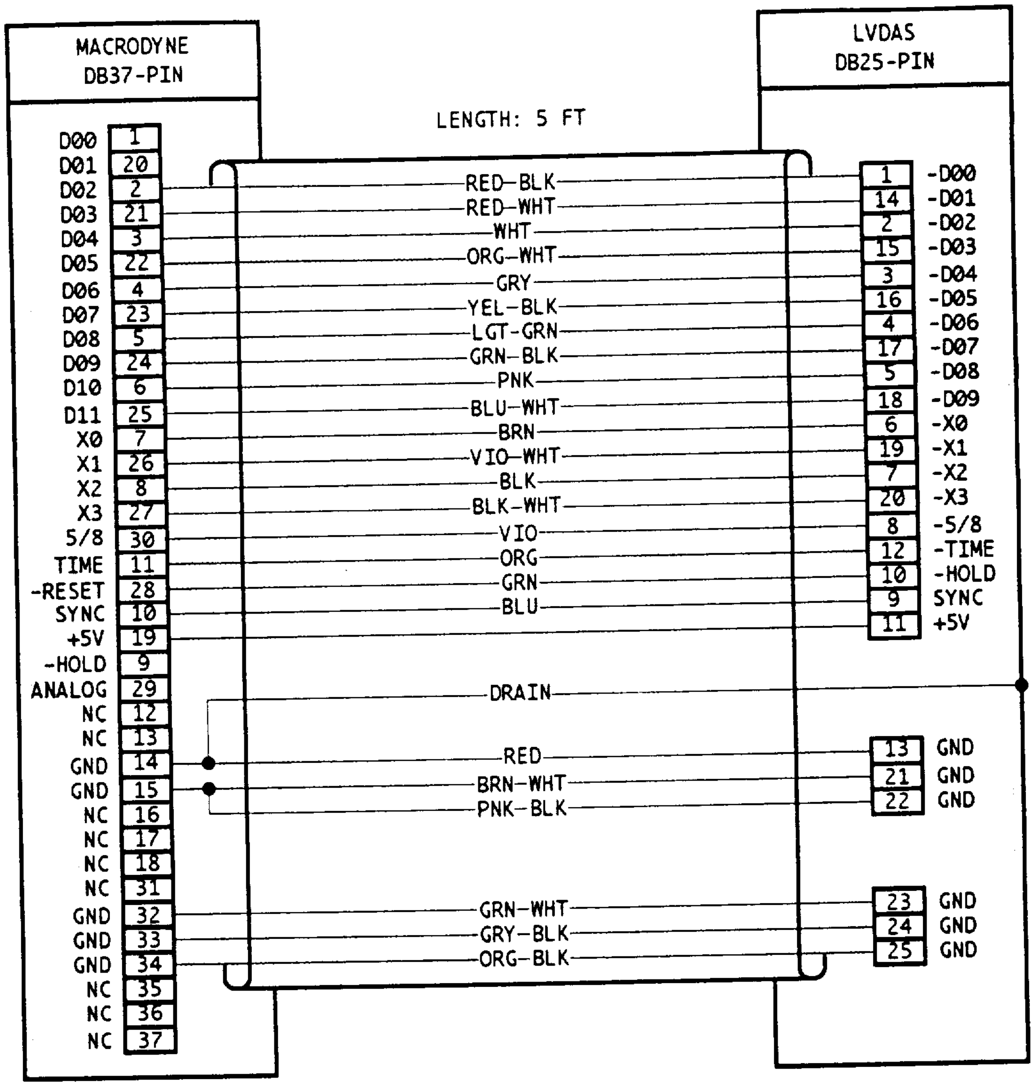


Figure 6. New Macrodyne to LVDAS Interface Cable Schematic Drawing.

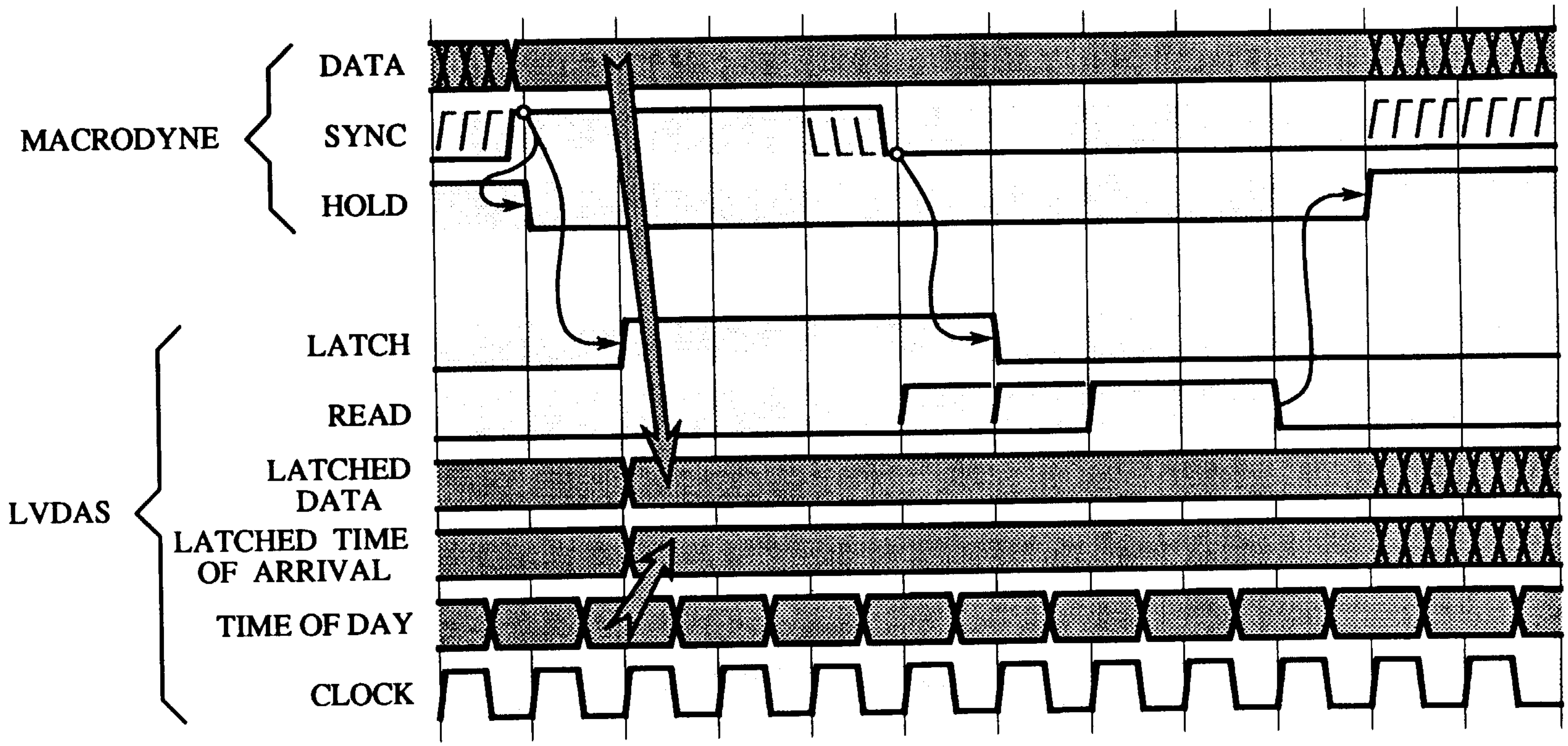


Figure 7. Macrodyne to LVDAS Interface Handshake Timing Diagram.

CHAPTER 3

DATA ACQUISITION COMPUTER HARDWARE AND SOFTWARE.

CHAPTER 3

Data Acquisition Computer Hardware and Software.

TABLE OF CONTENTS

LIST OF FIGURES

1.0 DATA ACQUISITION COMPUTER HARDWARE DESCRIPTION.

A simplified schematic drawing of the computer hardware is shown in Figure 1. The Figure also shows the how computer hardware interconnects with the Traverse Control System (TCS8) and the Laser Velocimeter Data Acquisition System (LVDAS). The data acquisition, data reduction, and data presentation computer system hardware was comprised of the following elements:

1. A Hewlett-Packard Series 9000 Model 375 Computer.
2. A System Interface Board.
3. A General Purpose Input/Output High Speed Interface.
4. Integral Hard Disk and Floppy Disk Drives.
5. Paint Jet Printer.

1.1 Hewlett-Packard Series 9000 Model 375 Computer.

The HP Series 9000 Model 375 computer was used to control the traverse system, acquire LDV data, perform data reduction and analysis, present the reduced data in graphical form, and to store the raw and reduced data on hard disk.

1.2 System Interface Board.

The system interface board possesses multiple serial and parallel interfaces. A normal IEEE-488 HPIB interface is used to send data to the Paint Jet printer. A high speed IEEE-488 HPIB interface is used to read data from and write data to the integral 40MByte Hard Disk and Floppy Disk Drives. The RS-232 serial interface is used to send commands as well as to send and receive position information from the Traverse Control System (TCS8).

1.3 General Purpose Input/Output High Speed Interface.

The General Purpose Input/Output (GPIO) High Speed Parallel Interface is used to send commands to the Laser Velocimeter Data Acquisition System (LVDAS). The LVDAS subsequently transmits back LDV data over this GPIO interface to the HP 9000-375 Computer.

1.4 Integral Hard Disk and Floppy Disk Drives.

The hard disk is partitioned into volumes. One volume contains system related files and the data acquisition program. The system files include the BASIC operating system and initialization programs that configure the computer, CRT display, and keyboard. The data acquisition program, which also resides on this volume, is automatically loaded and executed as part of the computers "power up" sequence. Another volume is used to store raw and reduced data for archival purposes and for future data reduction and analysis.

1.5 Paint Jet Printer.

The Paint Jet printer is used for listing programs and to print reduced data in tabular form. Additionally, graphs are "dumped" to provide a hard copy of histogram and profile plots.

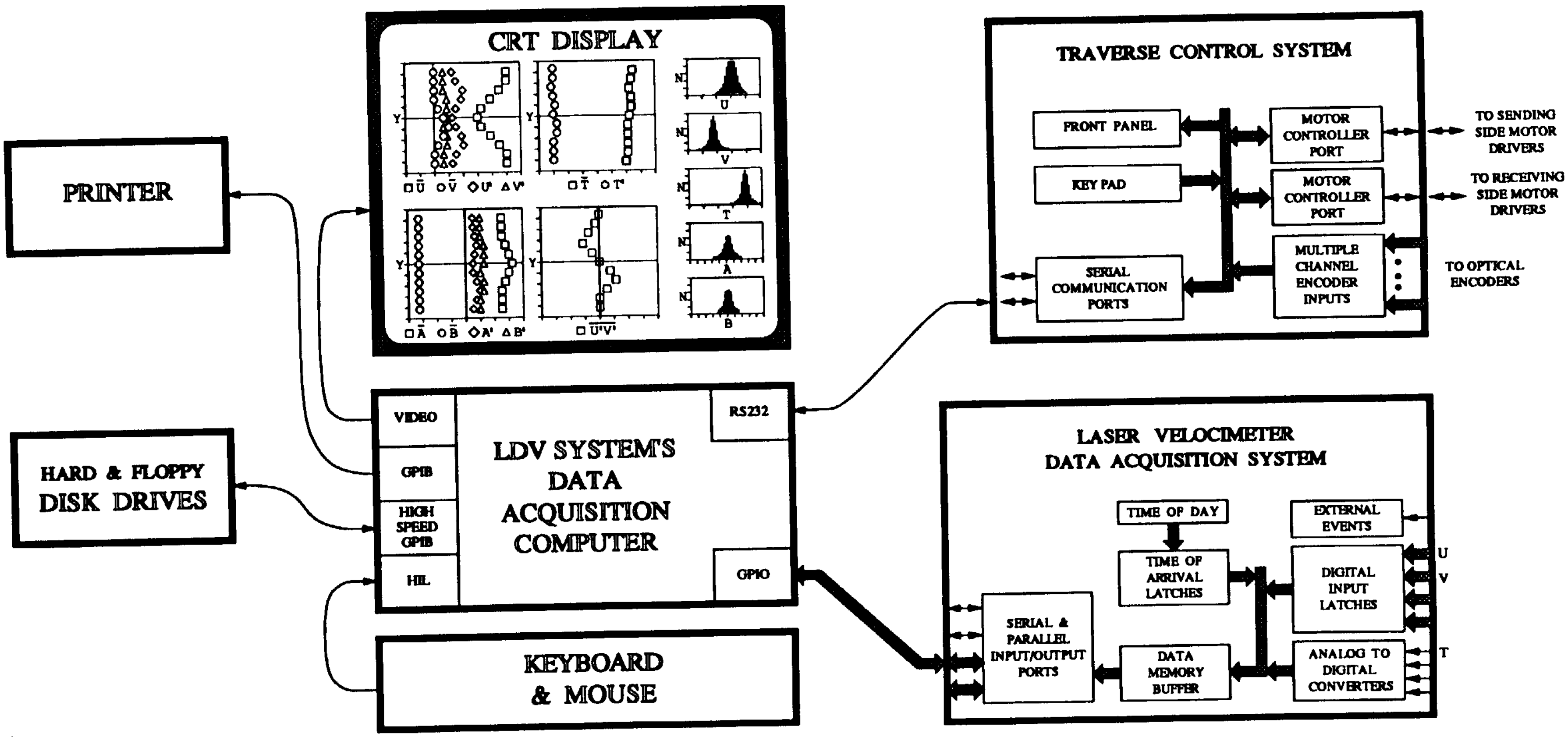


Figure 1. Laser Velocimeter System Configuration.

2.0 DATA ACQUISITION COMPUTER SOFTWARE DESCRIPTION.

The software used to control the traverse system and acquire tunnel data is listed in Appendix A and Appendix B of this report. Appendix A contains a catalog of the back-up floppy disk containing the system files and the data acquisition program named "3.5'HWT91". The system file "SYSB60" contains the BASIC 6.0 operating system. The "AUTOST" program is loaded and executed as part of the computer's "power up" sequence. This "AUTOST" program sets default values for the CRT and keyboard and then automatically loads and executes the "3.5'HWT91" program. Appendix A also contains a hardcopy listing of this "3.5'HWT91" program. This program is the original program that was used to acquire data during the hypersonic wind tunnel testing.

Appendix B is essentially a revised version with documentation of the program in Appendix A. The documentation is integrated into the code and includes information on how to boot the system and software. How to operate the menu driven software is also described. The documentation for the main program, each sub-routine, and each sub-program includes a description of the software, its purpose, and a list of variables with definitions. Appendix B contains a catalog of the back-up floppy disk containing the system files and the data acquisition program named "3.5'HWT92". The system file "SYSB60" contains the BASIC 6.0 operating system. The "AUTOST" program is loaded and executed as part of the computer's "power up" sequence. This "AUTOST" program sets default values for the CRT and Keyboard and then automatically loads and executes the "3.5'HWT92" program. Appendix B also contains a hardcopy listing of this "3.5'HWT92" program.

The following parts of Section 2 of this chapter contain a brief description of the data reduction applied to the raw data acquired by the "3.5'HWT" programs. A more complete set of documentation on the data reduction as well as coordinate system transformations can be found in Appendix C. Other topics are documented within the software code listing itself (refer to Appendix B for this software code listing.)

2.1 Instantaneous Velocities, Voltages, and Temperatures.

The following are the instantaneous velocities (U_i, V_i) which are derived from the digital data outputted from the Macrodyne counter signal processors. All velocities are measured in meters/second (m/s).

U_i : Instantaneous Streamwise Velocity.

V_i : Instantaneous Vertical Velocity.

The following are the instantaneous voltages (A_i, B_i) for the first two analog channels. All analog inputs are measured in volts (v).

A_i : Instantaneous Voltage on Analog Channel #1.

B_i : Instantaneous Voltage on Analog Channel #2.

The following is the stagnation temperature (T_i) which is inputted from the first analog channel (A_i). The temperatures are measured in degrees Rankine (°R).

T_i : Instantaneous Stagnation Temperature.

2.2 Velocity, Voltage, and Temperature Averages.

The instantaneous velocities (U_i, V_i), the instantaneous voltages (A_i, B_i), and the instantaneous stagnation temperatures (T_i) are summed so that the average velocities ($\overline{U}$, $\overline{V}$), the average voltages ($\overline{A}$, $\overline{B}$), and the average stagnation temperature ($\overline{T}$) can be calculated. All velocities are measured in meters/second (m/s), all analog inputs are measured in volts (v), and the stagnation temperature is measured in degrees Rankine (°R).

$\overline{U}$: Average Velocity (Streamwise).

$\overline{V}$: Average Velocity (Vertical).

$\overline{A}$: Average Voltage (Analog Channel #1).

$\overline{B}$: Average Voltage (Analog Channel #2).

$\overline{T}$: Average Stagnation Temperature.

$$\overline{U} = \frac{\sum_{i=1}^{n} [U_i]}{n} \qquad \text{m/s}$$

$$\overline{V} = \frac{\sum_{i=1}^{n} [V_i]}{n} \qquad \text{m/s}$$

$$\overline{A} = \frac{\sum_{i=1}^{n} [A_i]}{n} \qquad \text{v}$$

$$\overline{B} = \frac{\sum_{i=1}^{n} [B_i]}{n} \qquad \text{v}$$

$$\overline{T} = \frac{\sum_{i=1}^{n} [T_i]}{n} \qquad {}^{\circ}\text{R}$$

2.3 Velocity, Voltage, and Temperature Standard Deviations.

The velocity (U', V'), voltage (A', B'), and stagnation temperature (T') standard deviations are defined as shown here:

U' : Velocity Standard Deviation (Streamwise).
V' : Velocity Standard Deviation (Vertical).
A' : Voltage Standard Deviation (Analog Channel #1).
B' : Voltage Standard Deviation (Analog Channel #2).
T' : Stagnation Temperature Standard Deviation.

The following equations can be used to calculate the velocity (U', V'), voltage (A', B'), and stagnation temperature (T') standard deviations:

$$U' = \sqrt{\frac{\sum_{i=1}^{n} [U_i - \overline{U}]^2}{n}} \qquad \text{m/s}$$

$$V' = \sqrt{\frac{\sum_{i=1}^{n} [V_i - \overline{V}]^2}{n}} \qquad \text{m/s}$$

$$A' = \sqrt{\frac{\sum_{i=1}^{n} \left[A_i - \overline{A}\right]^2}{n}} \qquad \text{v}$$

$$B' = \sqrt{\frac{\sum_{i=1}^{n} \left[B_i - \overline{B}\right]^2}{n}} \qquad \text{v}$$

$$T' = \sqrt{\frac{\sum_{i=1}^{n} \left[T_i - \overline{T}\right]^2}{n}} \qquad {}^\circ\text{R}$$

The above equations are simplified to produce the following equations. The instantaneous velocities (U_i, V_i), voltages (A_i, B_i), and temperatures (T_i) are summed so that velocity (U', V'), voltage (A', B'), and stagnation temperature (T') standard deviations can be calculated. All velocity standard deviations are measured in meters/second (m/s), all voltage standard deviations are measured in volts (v), and all temperatures are measured in degrees Rankine (°R).

$$U' = \sqrt{\frac{\sum_{i=1}^{n} \left[U_i^2\right]}{n} - \overline{U}^2} \qquad \text{m/s}$$

$$V' = \sqrt{\frac{\sum_{i=1}^{n} \left[V_i^2\right]}{n} - \overline{V}^2} \qquad \text{m/s}$$

$$A' = \sqrt{\frac{\sum_{i=1}^{n} \left[A_i^2\right]}{n} - \overline{A}^2} \qquad \text{v}$$

$$B' = \sqrt{\frac{\sum_{i=1}^{n} \left[B_i^2\right]}{n} - \overline{B}^2} \qquad \text{v}$$

$$T' = \sqrt{\frac{\sum_{i=1}^{n}\left[T_i^2\right]}{n} - \overline{T}^{\,2}} \qquad {}^\circ R$$

The equations are simplified to these forms so that the software can compute summations of the instantaneous velocities, voltages, and temperature as well as the summations of their squares within the same software loop. This eliminates the need to calculate the difference values ($U_i-\overline{U}$, $V_i-\overline{V}$, $A_i-\overline{A}$, $B_i-\overline{B}$, $T_i-\overline{T}$). Also, the need to calculate the averages before the squared summations is removed.

2.4 Velocity, Voltage, and Temperature Cross Correlations.

The velocity:velocity shear stress ($\overline{U'V'}$), velocity:voltage cross correlations ($\overline{U'A'}$, $\overline{V'A'}$), and voltage:voltage cross correlations ($\overline{A'B'}$) are defined as shown here:

$\overline{U'V'}$: Velocity:Velocity Shear Stress.
$\overline{U'A'}$: Velocity:Voltage Cross Correlation.
$\overline{V'A'}$: Velocity:Voltage Cross Correlation.
$\overline{A'B'}$: Voltage:Voltage Cross Correlation.

The following equations can be used to calculate the shear stress and the cross correlations ($\overline{U'V'}$, $\overline{U'A'}$, $\overline{V'A'}$, $\overline{A'B'}$):

$$\overline{U'V'} = \frac{\sum_{i=1}^{n}\left[\left(U_i-\overline{U}\right)\left(V_i-\overline{V}\right)\right]}{n} \qquad m^2/s^2$$

$$\overline{U'A'} = \frac{\sum_{i=1}^{n}\left[\left(U_i-\overline{U}\right)\left(A_i-\overline{A}\right)\right]}{n} \qquad mv/s$$

$$\overline{V'A'} = \frac{\sum_{i=1}^{n} \left[(V_i - \overline{V})(A_i - \overline{A})\right]}{n} \qquad \text{mv/s}$$

$$\overline{A'B'} = \frac{\sum_{i=1}^{n} \left[(A_i - \overline{A})(B_i - \overline{B})\right]}{n} \qquad \text{v}^2$$

The above equations are simplified to produce the following equations. Summations of the instantaneous velocity and voltage (U_i, V_i, A_i, B_i) products are summed so that velocity:velocity shear stress ($\overline{U'V'}$), velocity:voltage cross correlations ($\overline{U'A'}$, $\overline{V'A'}$), and voltage:voltage cross correlation ($\overline{A'B'}$) can be calculated. All velocity:velocity shear stresses are measured in meters^2/second^2 (m^2/s^2). All velocity:voltage cross correlations are measured in meters•volts/second (mv/s). All voltage:voltage cross correlations are measured in volts^2 (v^2).

$$\overline{U'V'} = \frac{\sum_{i=1}^{n} [U_i V_i]}{n} - \overline{U}\ \overline{V} \qquad \text{m}^2/\text{s}^2$$

$$\overline{U'A'} = \frac{\sum_{i=1}^{n} [U_i A_i]}{n} - \overline{U}\ \overline{A} \qquad \text{mv/s}$$

$$\overline{V'A'} = \frac{\sum_{i=1}^{n} [V_i A_i]}{n} - \overline{V}\ \overline{A} \qquad \text{mv/s}$$

$$\overline{A'B'} = \frac{\sum_{i=1}^{n} [A_i B_i]}{n} - \overline{A}\ \overline{B} \qquad \text{v}^2$$

The equations are simplified to this form so that the software can compute summations of the instantaneous velocities and voltages (U_i, V_i, A_i, B_i) as well as the summations of their products within the same software loop. This eliminates the need to calculate the difference values ($U_i-\overline{U}$, $V_i-\overline{V}$, $A_i-\overline{A}$, $B_i-\overline{B}$). Also, the need to calculate the averages before the product summations is removed.

3.0 HP 98622A GPIO CARD TO LVDAS 16BIT PARALLEL I/O INTERFACE.

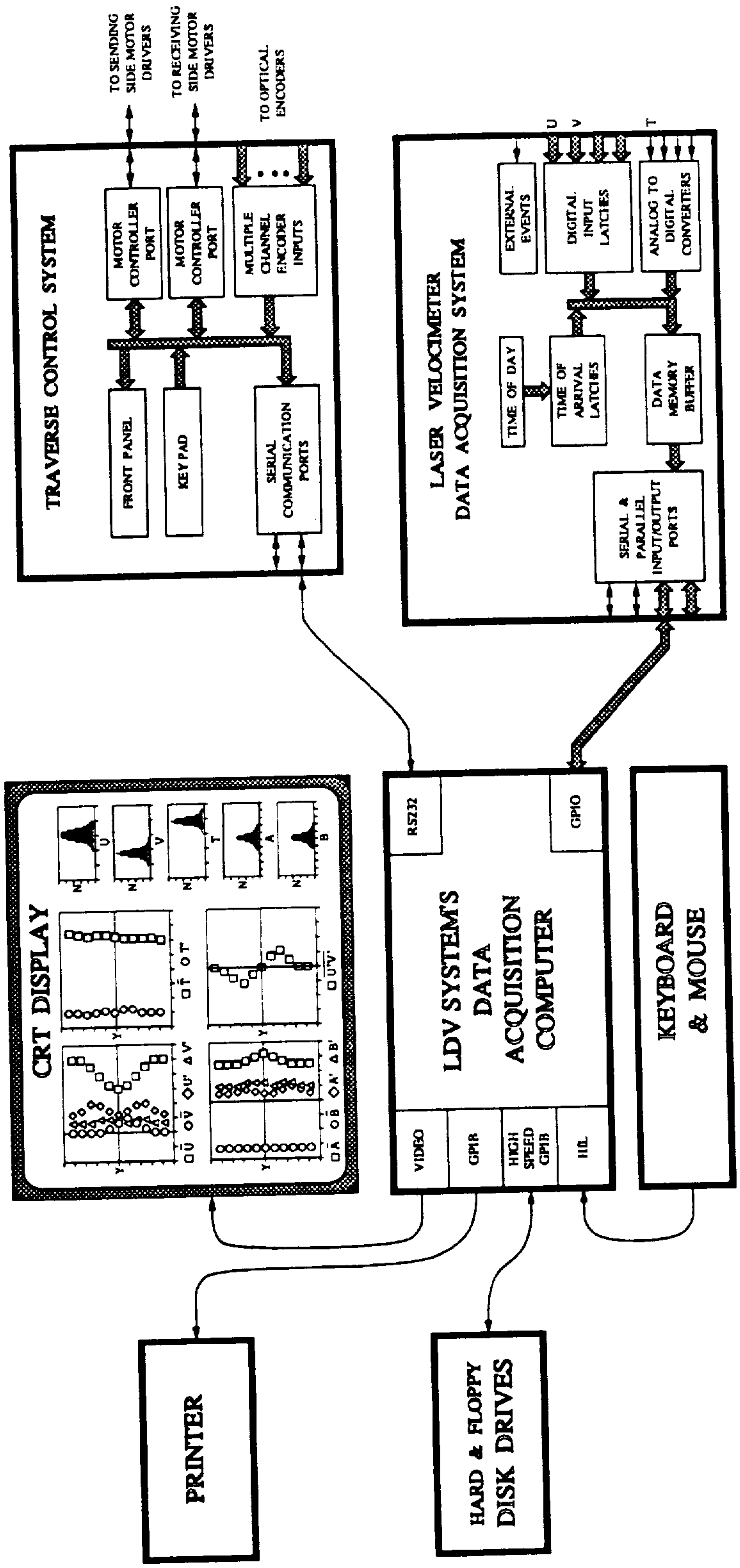
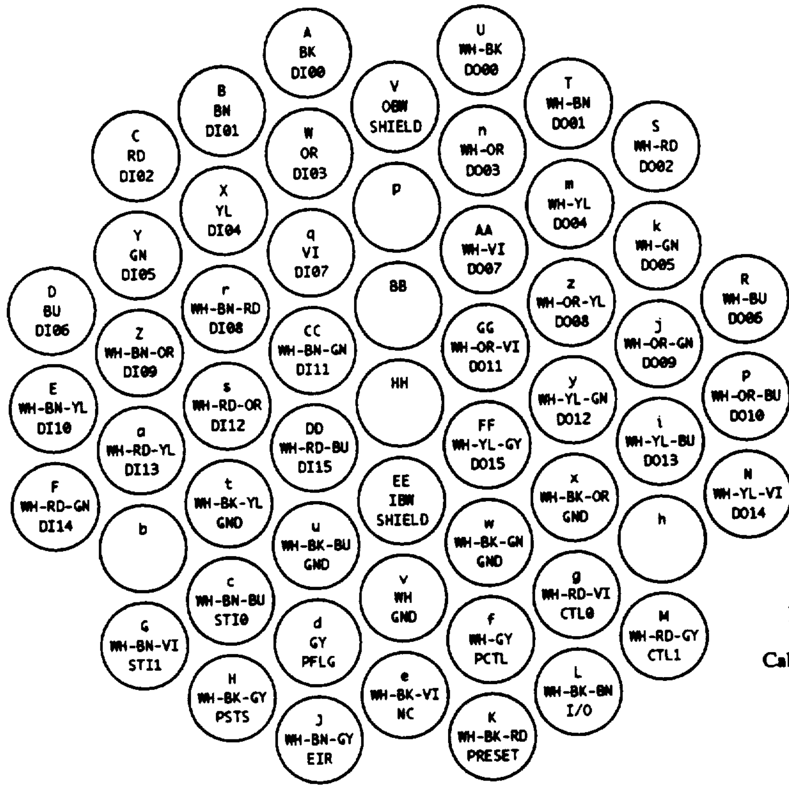
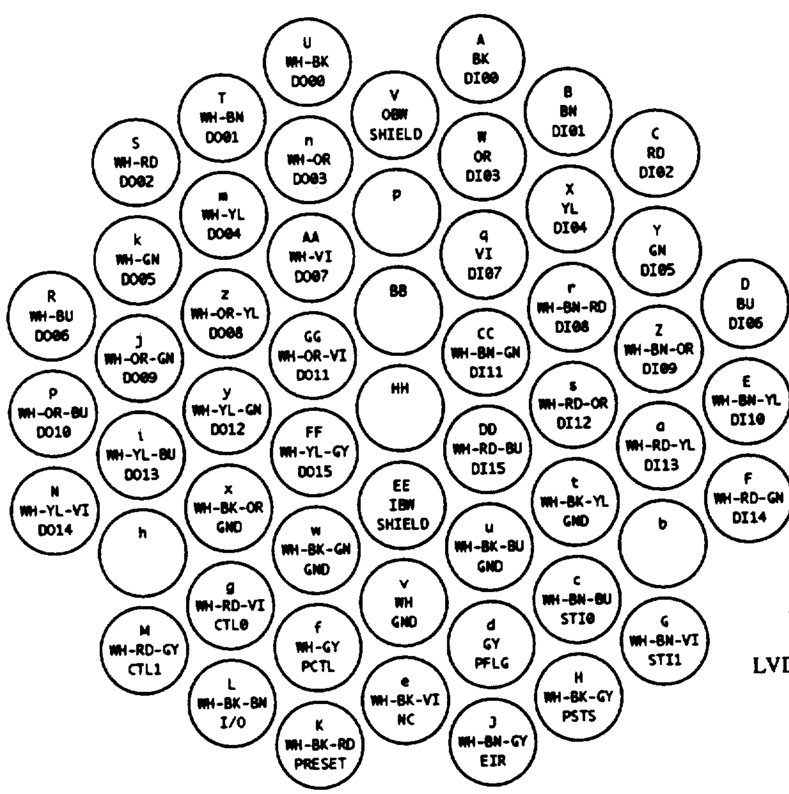


Figure 1. Laser Velocimeter System Configuration.



Front View
of
Cable Connector.



Front View
of
LVDAS Connector.

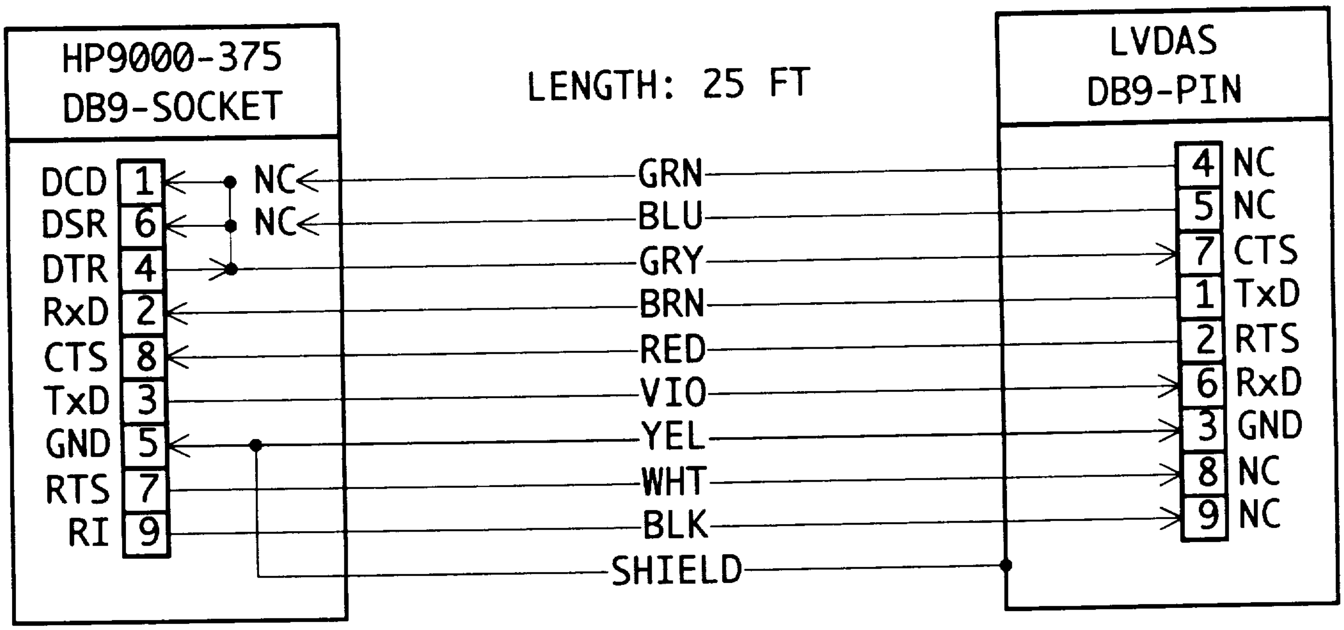
Figure 3. LVDAS Interface Circular Connector Pinout Positions.

HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface	HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface
HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface	HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface
HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface	HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface
HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface	HP9000 MODEL 3XX GPIO Interface to COMPLERE LVDAS Parallel Interface

Figure 4. HP Series 9000 Model 3xx to LVDAS Interface Cable Labels.

4.0 HP 9000 MODEL 375 TO LVDAS RS-232 SERIAL I/O INTERFACE.

HP Series 9000 Model 375
to
LVDAS
Serial Interface Cable



Cable Label

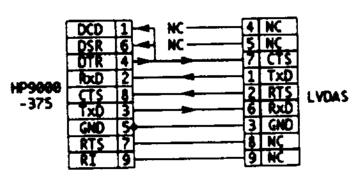
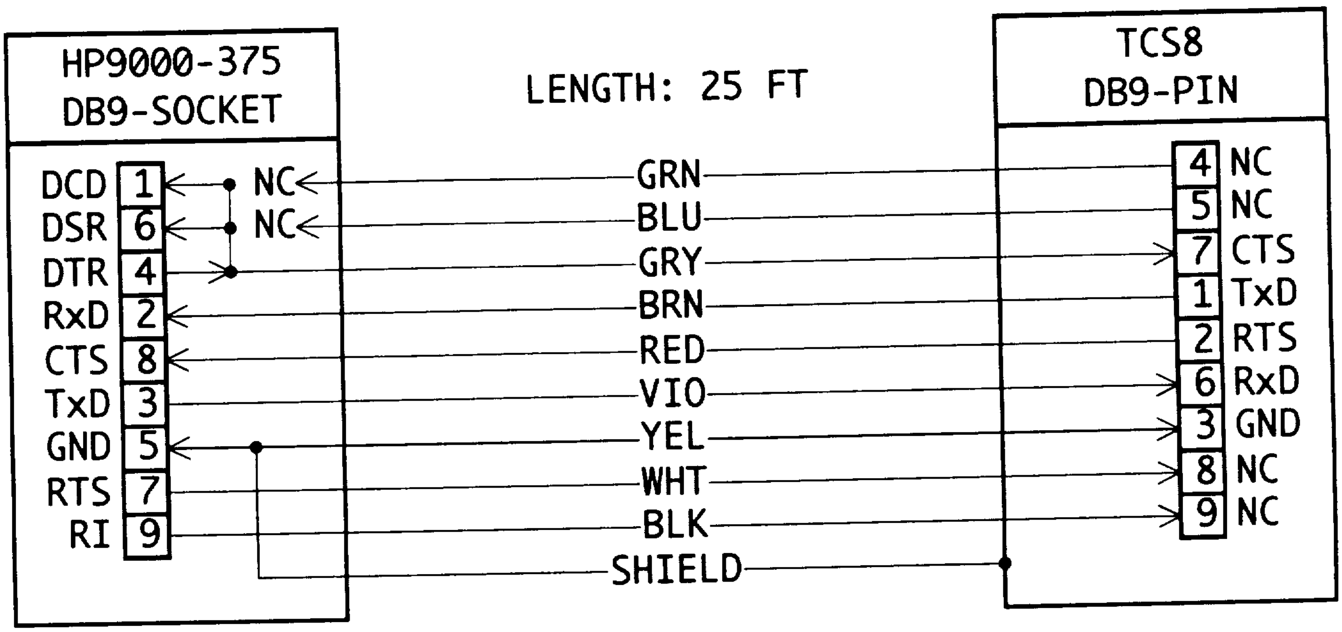


Figure 5. HP Series 9000 Model 375 to LVDAS Serial Interface Cable.

5.0 HP 9000 MODEL 3XX TO TCS8 RS-232 SERIAL I/O INTERFACE.

HP Series 9000 Model 375
to
TCS8
Serial Interface Cable



Cable Label

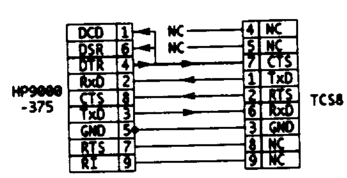


Figure 6. HP Series 9000 Model 375 to TCS8 Serial Interface Cable.

CHAPTER 4

TRAVERSE CONTROL SYSTEM.

CHAPTER 4

Traverse Control System.

TABLE OF CONTENTS

LIST OF FIGURES

1.00 THE TRAVERSE CONTROL SYSTEM

The traverse control system is made up of four sub-systems, see Fig. 1. The first sub-system is the main data taking computer (host computer). The second sub-system, the TCS8 (Traverse Control System 8 Axis), receives high level traverse commands from the host computer. The full duplex serial communications that links these two sub-systems allows the host computer to monitor the position and status of each axis in the system, see Section 4.00 Serial Interface Command Descriptions of the TCS8. The TCS8 can also function as a "stand alone" traverse controller. Through the use of the TCS8's front panel, an operator can execute all of the commands that the host computer can in addition the operator can control all axes in jog mode, see Section 2.00 Front Panel Descriptions of the TCS8 and Section 3.00 Local Command Descriptions of the TCS8. The third sub-system, the MDS (Motor Drive System), is controlled solely by the TCS8. The TCS8 translates the high level commands from the host computer and its front panel into low level indexer commands, see The Compumotor AX Drive User Manual. The TCS8 also receives encoder pulses from the traverses via the MDS. This allows the TCS8 to display real time position information on its front panel. The fourth and final sub-system of the traverse control system is the slide, motor, encoder, and limit switches that make up each axis. A drawing of each cable which is used to connect the traverse control system is included in Section 5.00 Traverse Control System Cables.

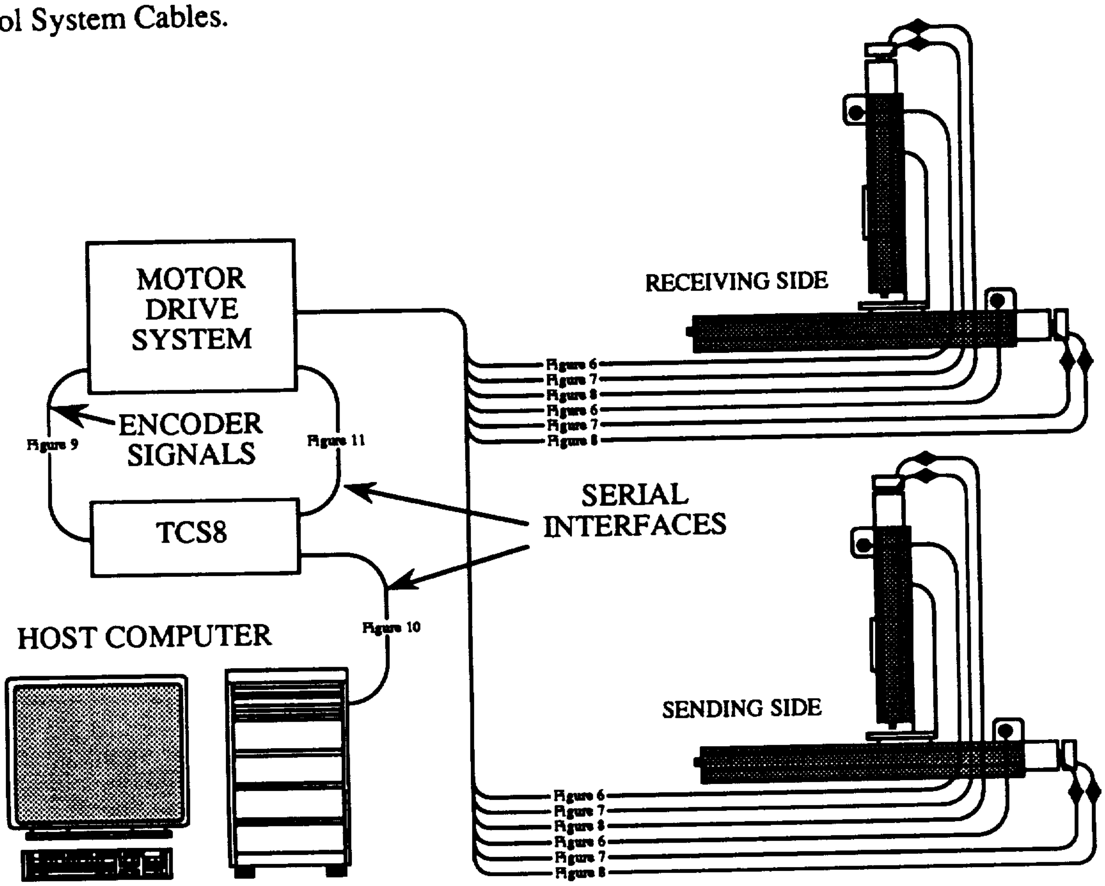


Figure 1 NASA Ames 3.5' HWT Traverse Control System.

1.01 The TCS8

The TCS8 is a microprocessor controlled system designed to interface an operator with a traverse system. The operator can utilize the TCS8 through the front panel, see Section 2.00 TCS8's Front Panel Descriptions and Section 3.00 TCS8's Local Command Descriptions, and/or with one or two host computers over serial interfaces, see Section 4.00 TCS8's Serial Interface Command Descriptions. The TCS8 stores all the critical parameters of motion, for each of the eight axes that it controls, in non-volatile memory. The critical parameters of motion being: position, encoder counts per unit travel, encoder counts per motor revolution, velocity, and acceleration. All of these parameters may be viewed, set, and saved. The TCS8 has three modes of motion; absolute, relative, and jog. With absolute movements, the operator specifies the final location. With relative movements, a distance is specified. With jogged movements, the operator presses a jog key on the front panel of the TCS8 until the desired location is obtained.

1.02 The Motor Drive System

There are four indexer/drivers used in this system. The TCS8 communicates with the indexers in the MDS's over a closed loop serial daisy chain. The 4/8 switch is located on the back panel of the MDS and must be set to 4, see Fig. 2. This figure also shows the location of all the motor, limit, and encoder connections. Channels X1, X2, Y1, and Y2 of the TCS8 control axis 1 through 4 on the first MDS. The TCS8 Encoders connector on the back of each MDS has a corresponding connector of the back of the TCS8, see Fig. A3 Schematic of TCS8 Back Panel. The interconnecting cable is detailed in Section 5.00 Traverse Control System Cables.

1.03 Positioning Resolution

The indexer/drivers that are used in the MDS can drive the motors at 12,800 steps/revolution. The encoders used on each axis are 100 pulses/revolution with quadrature encoding. Quadrature encoding adds a factor of 4 to the number of pulses/revolution to make this number 400 pulses/revolution. The final factor in the product of the resolution of an axis is the number of threads/inch of the lead screw. All of the axes of the traverse system have lead screws of 10 threads/inch. Thus, the positioning resolution of the axes with a 10 threads/inch lead screw is 0.00025 inches.

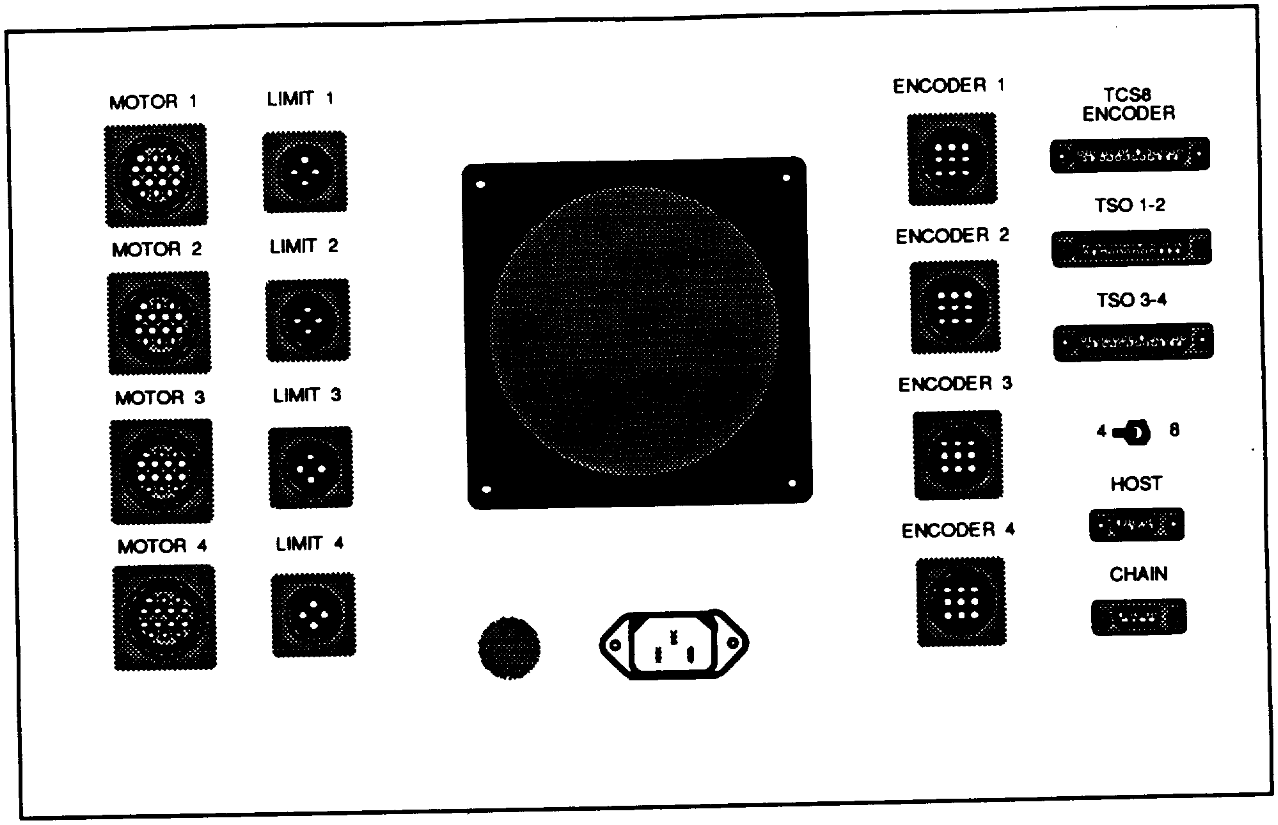


Figure 2 Schematic of Motor Drive System Back Panel.

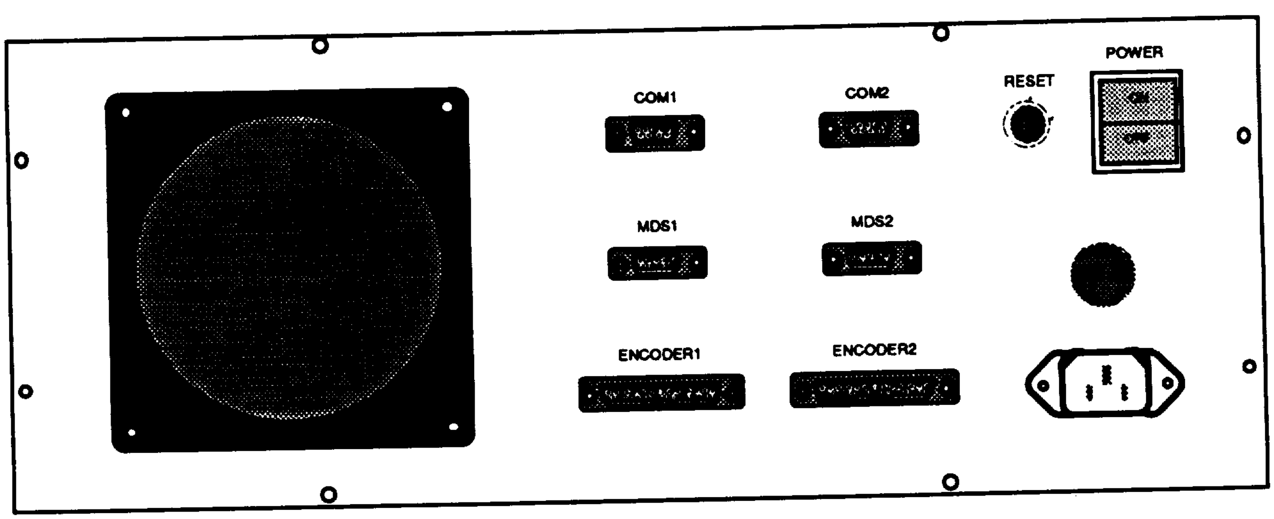


Figure 3 Schematic of TCS8 Back Panel.

1.04 Tunnel Penetration of Traverse Cables

The traverse slides for the 3.5' LDV System are located inside the pressurized test chamber and the traverse electronics are located outside, in the control room (see Fig 1. of Chapter 1). The traverse cables are fed through an existing access port on the north-east side of the test chamber. A special plate was designed (see Fig. 4) to replace an existing one. Bulkhead cable clamps are used to seal around the cables. When tightened down, these cable clamps compress a rubber grommet to create a seal. The four encoder cables are fed through one cable clamp and the four limit switch cables through another. The four motor cables, which are a larger diameter, are each fed through their own cable clamp.

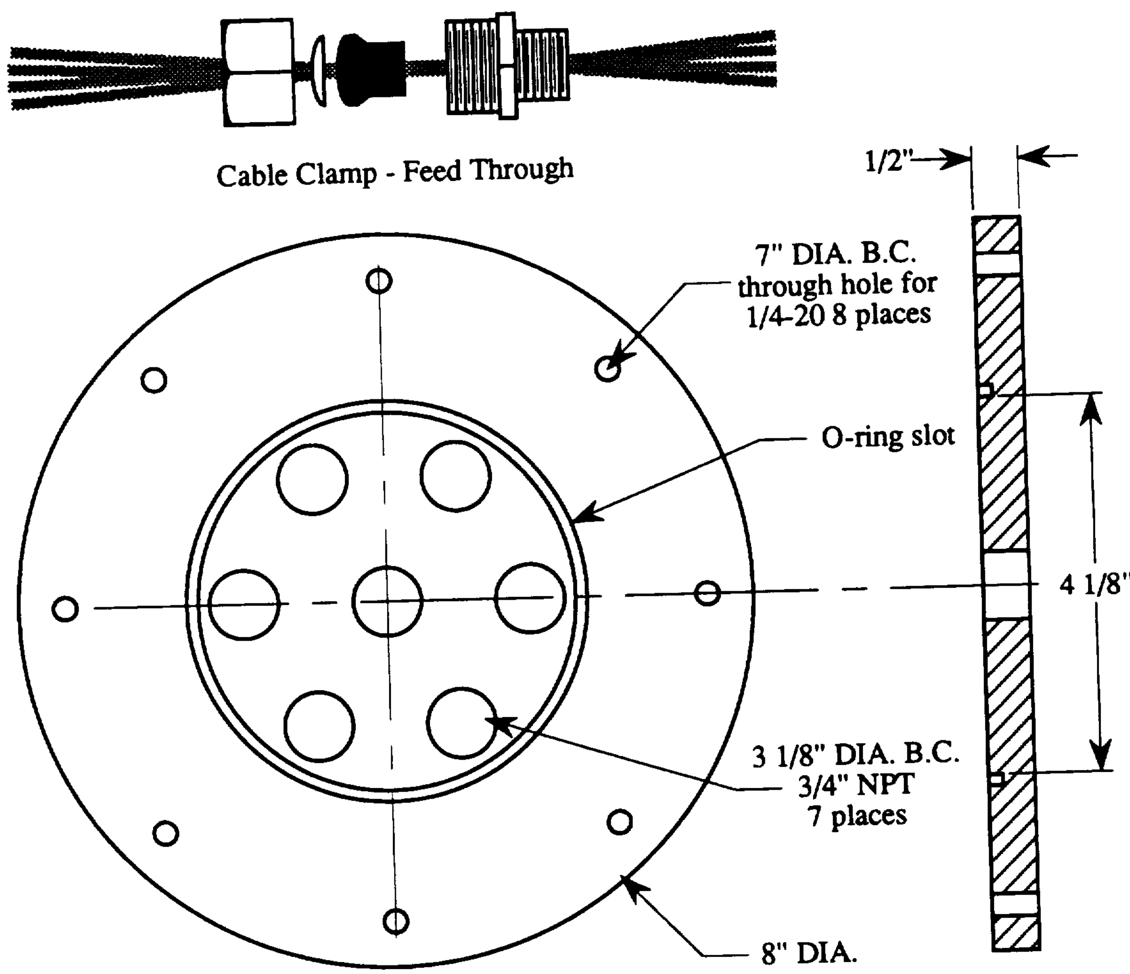


Figure 4 Tunnel Penetration Plate.

2.00 FRONT PANEL DESCRIPTION OF THE TCS8

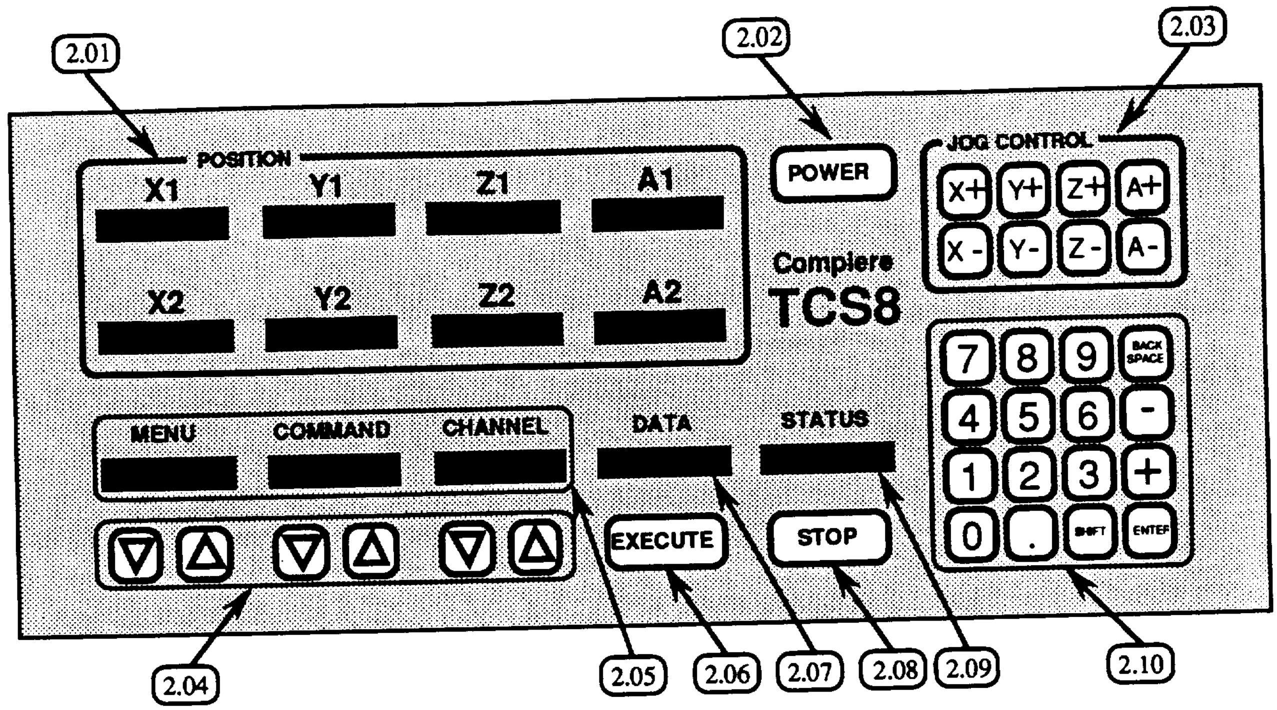


Figure 5 The Front Panel.

2.01 Position Display Windows.

There are eight windows corresponding to the eight axes that the TCS8 is capable of controlling. The position of each axis is continuously updated by monitoring its encoder, and displayed in a fixed format of a sign, two digits, a decimal point, and four digits.

2.02 Power Key.

The power key is used to store the current configuration to non-volatile memory before turning off power to the TCS8. Pressing the power key turns the displays off and saves the current configuration. Pressing it again turns the displays back on. This key can be used to implement a screen saver function.

2.03 Jog Control Keys.

These keys are used to control up to eight axes in a jog mode. The mode (slaved, one's only, or two's only) can be set through the jog menu. When the operator presses a jog key, the respective axis will begin to move. The direction that the axis moves is determined by the operator pressing either a plus or minus jog key. A plus jog key will turn the lead screw in a clockwise direction (away from the motor), a minus jog key will turn it in the counter-clockwise direction (towards the motor). By releasing the jog key, the operator stops motion on that axis. Motion will also stop if the axis reaches the limit for the direction it is moving, or if the indexer determines that the axis has stalled.

2.04 Scroll Keys.

These keys are used to scroll items through the MENU, COMMAND, and CHANNEL windows. All of the menus, their commands, and channel variations will be detailed in Section 3.00.

2.05 Command Windows.

These three windows (MENU, COMMAND, and CHANNEL) are used, in tandem with their respective scroll keys, to formulate a command to be executed by the TCS8.

2.06 Execute Key.

This key is used to execute the command currently formulated in the MENU, COMMAND, and CHANNEL windows.

2.07 Data Window.

Many of the TCS8's commands require some added data, e.g. the distance to move. Data for these commands are entered from the numeric key pad on the lower right of the TCS8 into the DATA window. Only a valid real number can be entered into the DATA window. If the operator enters an invalid real number, the character that is invalid will flash until the operator presses backspace or a valid character.

2.08 Stop Key.

The stop key, when pressed, will stop motion on all axes. The TCS8 will not lose track of the position of any axis. A move command started by the host computer and stopped by the stop key will finish normally with the position being reported. The position reported is the instantaneous position when the stop key was pressed. The final position of the axis being moved could be different than what was reported, thus the host computer should read the position again after a panic stop.

2.09 Status Window.

The STATUS window reflects the result of all commands. For commands that are not instantaneous, this window displays a busy status and then when the command completes it displays a ready status. The results of all view commands are displayed in the STATUS window. The STATUS window also displays the activity over the COM interfaces. For example, when the command for viewing position is sent over the COM1 interface, the STATUS window will display "COM1 VP" and when the command is completed the window will display "COM1 vp".

2.10 Numeric Key Pad.

The numeric key pad is used to enter a number into the data window. The user may backspace in the window or clear (shift-backspace) the window.

3.00 LOCAL COMMAND DESCRIPTIONS OF THE TCS8

This section describes the command set that can be executed from the front panel of the TCS8. Using the up and down keys under the MENU, COMMAND, and CHANNEL windows, the operator can formulate a command and then execute it by pressing the EXECUTE key. Some commands require extra information to be entered into the DATA window through the use of the numeric key pad. Each description includes a list of related commands that should be referred to in order to enhance the operator's understanding of the command. Also, where applicable, the default setting is given.

3.01 **Move to Zero.**

MENU: MOVE

COMMAND: TO ZERO

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE TO ZERO command is an easy way to move some or all of the axes to the zero position. This command can also be accomplished with the MOVE ABSOLUTE command and a zero in the DATA window. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before reaching zero, the rest of its movement is aborted.

RELATED COMMANDS: MOVE ABSOLUTE, MOVE RELATIVE, INIT Drive ON.

3.02 **Move Absolute.**

MENU: MOVE

COMMAND: ABSOLUTE

CHANNELS: X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE ABSOLUTE command requires a position to be entered in the DATA window. This position and the current position of the axis is used to calculate the relative distance the axis must move. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before reaching the position entered in the DATA window, the rest of its movement is aborted.

RELATED COMMANDS: MOVE TO ZERO, MOVE RELATIVE, INIT Drive ON

3.03 Move Relative.

MENU: MOVE

COMMAND: RELATIVE

CHANNELS: X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The MOVE RELATIVE command requires a distance to be entered in the DATA window. This position is used to calculate the relative distance the axis must move. Before using this command, selected axes must be initialized with the INIT DRIVE ON command. This command can be canceled by pressing the STOP key. When the STOP key is pressed, all axes will stop immediately. If an axis encounters a limit before moving the distance entered in the DATA window, the rest of its movement is aborted.

RELATED COMMANDS: MOVE TO ZERO, MOVE ABSOLUTE, INIT Drive ON

3.04 Jog Mode.

MENU: JOG

COMMAND: MODE

CHANNELS: SLAVED, ONE'S, TWO'S

DESCRIPTION: The JOG MODE command sets the way the JOG keys operate. When SLAVED is the setting, both the one and two axis of the X, Y, Z, or A coordinate will move the same amount. When ONE'S is the setting, only the one axes of the X, Y, Z, or A coordinate will move. And finally, when TWO'S is the setting, only the two axes of the X, Y, Z, or A coordinate will move. The current mode is marked with an asterisk. After setting the jog mode, jogged movements can be made using the jog control keys. As with the other movement commands, the axis or axes that are to be jogged must be initialized with the INIT Drive ON command.

RELATED COMMANDS: INIT Drive ON

DEFAULT: SLAVED

3.05 Set Counts Per Unit.

MENU: SET

COMMAND: CPU

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CPU command allows the user to change the counts per unit travel. The CPU for an axis is determined by multiplying the encoder resolution (counts/revolution) by the lead screw resolution (revolutions/unit of travel). A units conversion can be added here to change, for example, from inches to centimeters. When the CPU for an axis is changed, the position is automatically converted. This command requires a value to be entered in the DATA window.

RELATED COMMANDS: SET CPR, SET POSITION

DEFAULT:

X1	4000	X2	4000
Y1	4000	Y2	4000
Z1	4000	Z2	4000
A1	4000	A2	4000

3.06 Set Counts Per Revolution.

MENU: SET

COMMAND: CPR

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CPR command allows the user to change the encoder counts per motor revolution. The CPR for an axis is determined by dividing the encoder resolution (counts/revolution) by the lead screws resolution (revolutions/unit of travel). The encoder counts per motor revolution entered in the DATA window, must be a positive integer.

RELATED COMMANDS: SET CPU

DEFAULT:

X1	400	X2	400
Y1	400	Y2	400
Z1	400	Z2	400
A1	400	A2	400

3.07 **Set Position.**

MENU: SET

COMMAND: POSITION

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET POSITION command allows the user to change the current position of an axis without moving the axis. For example, the present position may be identified as zero or four inches or some other value. The new position must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET CPU

3.08 **Set Velocity.**

MENU: SET

COMMAND: VELOCITY

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET VELOCITY command allows the user to change the maximum speed at which an axis will travel. The range of valid velocities is 0.002 to 50.000 revolutions per second. The default is 5 revs/sec. Some stepper motor configurations will stall above a certain speed. To verify that a stall occurred, use the VIEW STALL command. When a stall happens, reduce the current velocity setting and continue normal operations. The new velocity must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET ACCEL.

DEFAULT:	X1	5.000	X2	5.000
	Y1	5.000	Y2	5.000
	Z1	5.000	Z2	5.000
	A1	5.000	A2	5.000

3.09 Set Acceleration.

MENU: SET

COMMAND: ACCEL.

CHANNELS: ALL, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET ACCEL. command allows the user to change the maximum acceleration for an axis. The range of valid accelerations is 0.01 to 999.99 revolutions per second per second. The default is 5 revs/sec/sec. The new acceleration must be entered in the DATA window before executing the command.

RELATED COMMANDS: SET VELOCITY

DEFAULT:				
	X1	5.00	X2	5.00
	Y1	5.00	Y2	5.00
	Z1	5.00	Z2	5.00
	A1	5.00	A2	5.00

3.10 Set Currents On.

MENU: SET

COMMAND: CrntsOn

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CrntsOn command allows the user to turn the motor currents on. The motor current must be on for an axis to be moved. The information in the DATA window is ignored.

RELATED COMMANDS: SET CrntsOff

3.11 Set Currents Off.

MENU: SET

COMMAND: CrntsOff

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET CrntsOff command allows the user to power down motors when they will not be used for long periods of time. The information in the DATA window is ignored.

RELATED COMMANDS: SET CrntsOn

3.12 Set Inits On.

MENU: SET

COMMAND: INITS ON

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The SET INITS ON command allows the user to initialize the indexers without turning on the power to the motors. This command gives the indexers their velocity, acceleration, and counts per motor revolution information. The indexers must have this information before any movement can occur. This information needs only to be given once after powering up the system. The information in the DATA window is ignored.

RELATED COMMANDS: INIT Drive ON

3.13 View Counts Per Unit.

MENU: VIEW

COMMAND: Cnt/Unit

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW Cnt/Unit command displays the current setting of the encoder counts per unit travel parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPU

3.14 View Counts Per Revolution.

MENU: VIEW

COMMAND: Cnt/MRev

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW Cnt/MRev command displays the current setting of the encoder counts per motor revolution parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPR

3.15 View Velocity.

MENU: VIEW

COMMAND: VELOCITY

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW VELOCITY command displays the current setting of the velocity parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET VELOCITY

3.16 View Acceleration.

MENU: VIEW

COMMAND: ACCEL.

CHANNELS: X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The VIEW ACCEL. command displays the current setting of the acceleration parameter for the selected axis in the STATUS window. The information in the DATA window is ignored.

RELATED COMMANDS: SET ACCEL.

3.17 View Init.

MENU: VIEW

COMMAND: INIT

CHANNELS: none

DESCRIPTION: The VIEW INIT command uses the STATUS window to display a one(initialized) or a zero(uninitialized) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: SET INITS, INIT Drive ON

3.18 View Currents.

MENU: VIEW

COMMAND: CURRENTS

CHANNELS: none

DESCRIPTION: The VIEW CURRENTS command uses the STATUS window to display a one(current on) or a zero(current off) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. The information in the DATA window is ignored.

RELATED COMMANDS: SET CrntsOn, SET CrntsOff, INIT Drive ON, INIT Drive OFF

3.19 View Plus Limit Switches.

MENU: VIEW

COMMAND: Plus LMT

CHANNELS: none

DESCRIPTION: The VIEW Plus LMT command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. The plus limit switches are located at the positive movement end of travel. The information in the DATA window is ignored.

RELATED COMMANDS: VIEW Minus LMT, VIEW HOME

3.20 View Minus Limit Switches.

MENU: VIEW

COMMAND: Minus LMT

CHANNELS: none

DESCRIPTION: The VIEW Minus LMT command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. The minus limit switches are located at the negative movement end of travel. The information in the DATA window is ignored.

RELATED COMMANDS: VIEW Plus LMT, VIEW HOME

3.21 View Home Switches.

MENU: VIEW

COMMAND: HOME

CHANNELS: none

DESCRIPTION: The VIEW HOME command uses the STATUS window to display a one(on limit) or a zero(not on limit) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. The home limit switch can be adjusted by the user for application specific tasks. The information in the DATA window is ignored.

RELATED COMMANDS: VIEW Plus LMT, VIEW Minus LMT

3.22 View Stall Indication.

MENU: VIEW

COMMAND: STALL

CHANNELS: none

DESCRIPTION: The VIEW STALL command uses the STATUS window to display a one(stalled) or a zero(not stalled) for each axis. The STATUS window has eight characters; left to right respectively reflecting the status of X1, X2 ..., A1, A2. A stall is indicated when the indexer is making a move and the amount of pulses send to the motor does not match the corresponding number of pulses received from the encoder. A stall can occur if the velocity or acceleration is set to high, the encoder counts per motor revolution are set incorrectly, or the axis is physically jammed. The information in the DATA window is ignored.

RELATED COMMANDS: none

3.23 Init Default.

MENU: INIT

COMMAND: DEFAULT

CHANNELS: none

DESCRIPTION: The INIT DEFAULT command restores the initial factory defaults (CPU, CPR, VELOCITY, ACCELERATION,BAUD RATE,BITS/CHAR,PARITY,STOP BITS,HANDSHAKE) of the TCS8. After executing this command, execute the command INIT Drive ON to initialize the indexers. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPU, SET CPR, SET VELOCITY, SET ACCEL.

3.24 Init Drive On.

MENU: INIT

COMMAND: Drive ON

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The INIT Drive ON command initializes the selected axes for movement. This command does the same thing as SET INITS ON except that it also turns on the current to the motors. The information in the DATA window is ignored.

RELATED COMMANDS: SET CPU, SET CPR, SET VELOCITY, SET ACCEL., SET CnrtsOn, SET CnrtsOff, INIT DEFAULT

3.25 Init Drive Off.

MENU: INIT

COMMAND: Drive OFF

CHANNELS: ALL, ONE'S, TWO'S, X1&X2, Y1&Y2, Z1&Z2, A1&A2, X1, X2, Y1, Y2, Z1, Z2, A1, A2

DESCRIPTION: The INIT Drive OFF command is an alias for SET CrntsOff.

RELATED COMMANDS: SET CrntsOff

3.26 COM1/COM2 Baud Rate.

MENU: COM1/COM2

COMMAND: BaudRate

CHANNELS: 19.2K, 9600, 4800, 2400, 1200, 300, 110

DESCRIPTION: The COM1/COM2 BaudRate commands set the baud rate for the selected communication channel. The information in the DATA window is ignored. The current baud rate is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: 9600

3.27 COM1/COM2 Bits Per Character.

MENU: COM1/COM2

COMMAND: Bit/Char

CHANNELS: SEVEN, EIGHT

DESCRIPTION: The COM1/COM2 Bit/Char command set the bits per character for the selected communication channel. The information in the DATA window is ignored. The current number of bits per character is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: EIGHT

3.28 COM1/COM2 Parity.

MENU: COM1/COM2

COMMAND: Parity

CHANNELS: NONE, EVEN, ODD

DESCRIPTION: The COM1/COM2 Parity command set the parity for the selected communication channel. The information in the DATA window is ignored. The current parity is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: EVEN

3.29 COM1/COM2 Stop Bits.

MENU: COM1/COM2

COMMAND: StopBits

CHANNELS: 1, 1.5, 2

DESCRIPTION: The COM1/COM2 StopBits command set the stop bits for the selected communication channel. The information in the DATA window is ignored. The current number of stop bits is marked with an asterisk.

RELATED COMMANDS: none

DEFAULT: 1

3.30 COM1/COM2 Handshake.

MENU: COM1/COM2

COMMAND: HandShak

CHANNELS: NO, YES

DESCRIPTION: The COM1/COM2 HandShak command set the handshake for the selected communication channel. The information in the DATA window is ignored. An asterisk marks whether there is handshaking or not.

RELATED COMMANDS: none

DEFAULT: YES

4.00 SERIAL INTERFACE COMMAND DESCRIPTIONS OF THE TCS8

This section describes the command set that can be executed through the serial interfaces of the TCS8. Each description includes a code section that outlines the characters that must be sent to execute the command. The vertical bar in this section is used as a separator and is not sent as part of the command code. The symbol "CRLF" stands for the two characters carriage return and line feed. Also where applicable, the default setting is given.

4.01 Change Serial Configuration.

COMMAND: CHANGE SERIAL CONFIGURATION

CODE: CS COM;CATEGORY;ATTRIBUTE;

PARAMETERS:	COM:	1/COM1
		2/COM2
	CATEGORY:	0/BAUDRATE
	ATTRIBUTE:	0/19.2K
		1/9600
		2/4800
		3/2400
		4/1200
		5/300
		6/110
	CATEGORY:	1(BITS PER CHARACTER)
	ATTRIBUTE:	0/SEVEN
		1/EIGHT
	CATEGORY:	2(PARITY)
	ATTRIBUTE:	0/NONE
		1/EVEN
		2/ODD
	CATEGORY:	3(STOP BITS)
	ATTRIBUTE:	0/ONE
		1/ONE AND A HALF
		2/TWO
	CATEGORY:	4(HANDSHAKE)
	ATTRIBUTE:	0/NO
		1/YES

DESCRIPTION: This command must be executed with extreme caution and forethought. If the user changes an attribute of the same COM port from which he is sending the command, he must change to that attribute on the host computer before sending the next command. The best way to change the serial configuration of a COM port is to utilize the front panel commands.

DEFAULT: 9600 baud, EIGHT bits/char, EVEN parity, ONE stop bit, handshaking YES

EXAMPLE: To change the baudrate of COM1 to 2400 the user must send CS1;0;3;

4.02 **Move to Absolute Position.**

COMMAND: MOVE TO ABSOLUTE POSITION AND REPORT FINAL POSITION

CODE: MA CHANNEL:POSITION,CHANNEL:POSITION,...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2
	POSITION:	Real number free format

DESCRIPTION: This command moves selected channels to absolute positions.

EXAMPLES: To move all channels to zero the user may send MA0:0,CRLF or MA12345678:0,CRLF. To move channel X1 to zero the user must send MA1:0,CRLF. To move channels X1 and X2 to zero the user may send MA12:0,CRLF or MA1:0,2:0,CRLF or MA1:0,CRLF and MA2:0,CRLF.

4.03 **Move Relative to Current Position.**

COMMAND: MOVE TO RELATIVE DISTANCE AND REPORT FINAL POSITION

CODE: MR CHANNEL:DISTANCE,CHANNEL:DISTANCE,...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2
	POSITION:	Real number free format

DESCRIPTION: This command moves selected channels relative distances.

EXAMPLES: To move all channels one unit the user may send MR0:1,CRLF or MR12345678:1,CRLF. To move channel X1 one unit the user must send MR1:1,CRLF. To move channels X1 and X2 one unit the user may send MR12:1,CRLF or MR1:1,2:1,CRLF or MR1:1,CRLF and MR2:1,CRLF.

4.04 **Set Acceleration.**

COMMAND: SET ACCELERATION

CODE: SA CHANNEL:ACCELERATION,CHANNEL:ACCELERATION,...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2
	ACCELERATION:	Real number free format between 0.01 and 99.99 inclusive.

DESCRIPTION: This command sets the acceleration for selected channels.

DEFAULT: All channels 5.00 revolutions/second/second

EXAMPLES: To set the acceleration for all channels to 4.00 revolutions/second/second the user may send SA0:4.00,CRLF or SA12345678:4.00,CRLF. To set the acceleration for channel X1 to 4.00 revolutions/second/second the user must send SA1:4.00,CRLF. To set the acceleration for channels X1 and X2 to 4.00 revolutions/second/second the user may send SA12:4.00,CRLF or SA1:4.00 ,2:4.00,CRLF or SA1:4.00,CRLF and SA2:4.00,CRLF.

4.05 **View Acceleration.**

COMMAND: VIEW ACCELERATION

CODE: VA CHANNEL|CHANNEL...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2

DESCRIPTION: This command views the acceleration for selected channels. The TCS8 transmits each of the accelerations requested back to the host computer separated by carriage return line feeds.

EXAMPLES: To view the acceleration for all channels the user may send VA0CRLF or VA12345678CRLF. To view the acceleration for channel X1 the user must send VA1CRLF To view the acceleration for channels X1 and X2 the user may send VA12CRLF or VA1CRLF and VA2CRLF.

4.06 Set Velocity.

COMMAND: SET VELOCITY

CODE: SV CHANNEL:VELOCITY,CHANNEL:VELOCITY,...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

VELOCITY: Real number free format between 0.001 and 50.000 inclusive.

DESCRIPTION: This command sets the velocity for selected channels.

DEFAULT: All channels 5.000 revolutions/second

EXAMPLES: To set the velocity for all channels to 4.00 revolutions/second the user may send SV0:4.00,CRLF or SV12345678:4.00,CRLF. To set the velocity for channel X1 to 4.00 revolutions/second the user must send SV1:4.00,CRLF. To set the velocity for channels X1 and X2 to 4.00 revolutions/second the user may send SV12:4.00,CRLF or SV1:4.00 ,2:4.00,CRLF or SV1:4.00,CRLF and SV2:4.00,CRLF.

4.07 View Velocity.

COMMAND: VIEW VELOCITY

CODE: VV CHANNEL|CHANNEL...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the velocity for selected channels. The TCS8 transmits each of the velocities requested back to the host computer separated by carriage return line feeds.

EXAMPLES: To view the velocity for all channels the user may send VV0CRLF or VV12345678CRLF. To view the velocity for channel X1 the user must send VV1CRLF. To view the velocity for channels X1 and X2 the user may send VV12CRLF or VV1CRLF and VV2CRLF.

4.08 **Set Encoder Counts Per Unit of Travel.**

COMMAND: SET ENCODER COUNTS PER UNIT TRAVEL

CODE: SU CHANNEL:CPU,CHANNEL:CPU,...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

CPU: Non-zero real number free format.

DESCRIPTION: This command sets the encoder counts per unit travel for selected channels.

DEFAULT: X1,X2,Y1,Y2,Z1,Z2,A1, and A2 4000 counts/inch

EXAMPLES: To set the encoder counts per unit travel for all channels to 5000 the user may send SU0:5000,CRLF or SU12345678:5000,CRLF. To set the encoder counts per unit travel for channel X1 to 5000 the user must send SU1:5000,CRLF. To set the encoder counts per unit travel for channels X1 and X2 to 5000 the user may send SU12:5000,CRLF or SU1:5000 ,2:5000,CRLF or SU1:5000,CRLF and SU2:5000,CRLF.

4.09 **View Encoder Counts Per Unit of Travel.**

COMMAND: VIEW ENCODER COUNTS PER UNIT TRAVEL

CODE: VU CHANNEL|CHANNEL...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the encoder counts per unit travel for selected channels. The TCS8 transmits each of the encoder counts per unit travel requested back to the host computer separated by carriage return line feeds.

EXAMPLES: To view the encoder counts per unit travel for all channels the user may send VU0CRLF or VU12345678CRLF. To view the encoder counts per unit travel for channel X1 the user must send VU1CRLF. To view the encoder counts per unit travel for channels X1 and X2 the user may send VU12CRLF or VU1CRLF and VU2CRLF.

4.10 Set Counts Per Motor Revolution.

COMMAND: SET ENCODER COUNTS PER MOTOR REVOLUTION

CODE: SR CHANNEL:CPR,CHANNEL:CPR,...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

CPU: Non-zero integer free format.

DESCRIPTION: This command sets the encoder counts per motor revolution for selected channels.

DEFAULT: X1,X2,Y1,Y2,Z1,Z2 and A1,A2 400 counts/inch

EXAMPLES: To set the encoder counts per motor revolution for all channels to 500 the user may send SR0:500,CRLF or SR12345678:500,CRLF. To set the encoder counts per motor revolution for channel X1 to 500 the user must send SR1:500,CRLF. To set the encoder counts per motor revolution for channels X1 and X2 to 500 the user may send SR12:500,CRLF or SR1:500 ,2:500,CRLF or SR1:500,CRLF and SR2:500,CRLF.

4.11 View Counts Per Motor Revolution.

COMMAND: VIEW ENCODER COUNTS PER MOTOR REVOLUTION

CODE: VR CHANNEL|CHANNEL...|CRLF

PARAMETERS: CHANNEL: 0/ALL CHANNELS
1/X1
2/X2
3/Y1
4/Y2
5/Z1
6/Z2
7/A1
8/A2

DESCRIPTION: This command views the encoder counts per motor revolution for selected channels. The TCS8 transmits each of the encoder counts per motor revolution requested back to the host computer separated by carriage return line feeds.

EXAMPLES: To view the encoder counts per motor revolution for all channels the user may send VR0CRLF or VR12345678CRLF. To view the encoder counts per motor revolution for channel X1 the user must send VR1CRLF. To view the encoder counts per motor revolution for channels X1 and X2 the user may send VR12CRLF or VR1CRLF and VR2CRLF.

4.12 Set Position.

COMMAND: SET POSITION

CODE: SP CHANNEL:POSITION,CHANNEL:POSITION,...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2
	POSITION:	real number.

DESCRIPTION: This command sets the position for selected channels.

EXAMPLES: To set the position for all channels to 1.5 the user may send SP0:1.5,CRLF or SP12345678:1.5,CRLF. To set the position for channel X1 to 1.5 the user must send SP1:1.5,CRLF. To set the position for channels X1 and X2 to 1.5 the user may send SP12:1.5,CRLF or SP1:1.5 ,2:1.5,CRLF or SP1:1.5,CRLF and SP2:1.5,CRLF.

4.13 View Position.

COMMAND: VIEW POSITION

CODE: VP CHANNEL|CHANNEL...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS 1/X1 2/X2 3/Y1 4/Y2 5/Z1 6/Z2 7/A1 8/A2

DESCRIPTION: This command views the position for selected channels. The TCS8 transmits each of the positions requested back to the host computer separated by carriage return line feeds.

EXAMPLES: To view the position for all channels the user may send VP0CRLF or VP12345678CRLF. To view the position for channel X1 the user must send VP1CRLF. To view the position for channels X1 and X2 the user may send VP12CRLF or VP1CRLF and VP2CRLF.

4.14 **Set Current to Motor Windings.**

COMMAND: SET CURRENT TO MOTOR WINDINGS

CODE: SC CHANNEL:ON/OFF,CHANNEL:ON/OFF,...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS
		1/X1
		2/X2
		3/Y1
		4/Y2
		5/Z1
		6/Z2
		7/A1
		8/A2
	ON/OFF:	1/ON
		0/OFF

DESCRIPTION: This command sets the current to the motor windings for selected channels on or off.

EXAMPLES:
To set the current to the motor windings for all channels on the user may send SC0:1,CRLF or SC12345678:1,CRLF to set them off the user may send SC0:0,CRLF or SC12345678:0,CRLF.

4.15 **View Current to Motor Windings.**

COMMAND: VIEW CURRENT TO MOTOR WINDINGS

CODE: VC CHANNEL|CHANNEL...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS
		1/X1
		2/X2
		3/Y1
		4/Y2
		5/Z1
		6/Z2
		7/A1
		8/A2

DESCRIPTION: This command views the current to the motor windings for selected channels. The TCS8 transmits each response of on/off (1/0) back to the host computer separated by carriage return line feeds.

EXAMPLES:
To view the current to the motor windings for all channels the user may send VC0CRLF or VC12345678CRLF
To view the current to the motor windings for channel X1 the user must send VC1CRLF
To view the current to the motor windings for channels X1 and X2 the user may send VC12CRLF or VC1CRLF and VC2CRLF.

4.16 Set Initialization of Indexer/Drivers.

COMMAND: SET INITIALIZATION OF INDEXER/DRIVERS

CODE: SI CHANNEL|CHANNEL...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS
		1/X1
		2/X2
		3/Y1
		4/Y2
		5/Z1
		6/Z2
		7/A1
		8/A2

DESCRIPTION: This command sends the current value of the acceleration, velocity, and the encoder counts per motor revolution to the indexer/driver for the selected channels. This command must be sent before any move commands may be sent.

EXAMPLES:
To initialize all channels the user may send SI0CRLF or SI12345678CRLF
To initialize channel X1 the user must send SI1CRLF
To initialize channels X1 and X2 the user may send SI12CRLF or SI1CRLF and SI2CRLF

4.17 View Initialization of Indexer/Drivers.

COMMAND: VIEW INITIALIZATION OF INDEXER/DRIVERS

CODE: VI CHANNEL|CHANNEL...|CRLF

PARAMETERS:	CHANNEL:	0/ALL CHANNELS
		1/X1
		2/X2
		3/Y1
		4/Y2
		5/Z1
		6/Z2
		7/A1
		8/A2

DESCRIPTION: This command returns "1" if the indexer/driver has been initialized since the TCS8 was turned on and "0" if it has not. The TCS8 transmits each of the responses back to the host computer separated by carriage return line feeds.

EXAMPLES:
To check the initialization of all channels the user may send VI0CRLF or VI12345678CRLF
To check the initialization of channel X1 the user must send VI1CRLF
To check the initialization of channels X1 and X2 the user may send VI12CRLF or VI1CRLF and VI2CRLF

5.00 Traverse Control System Cables.

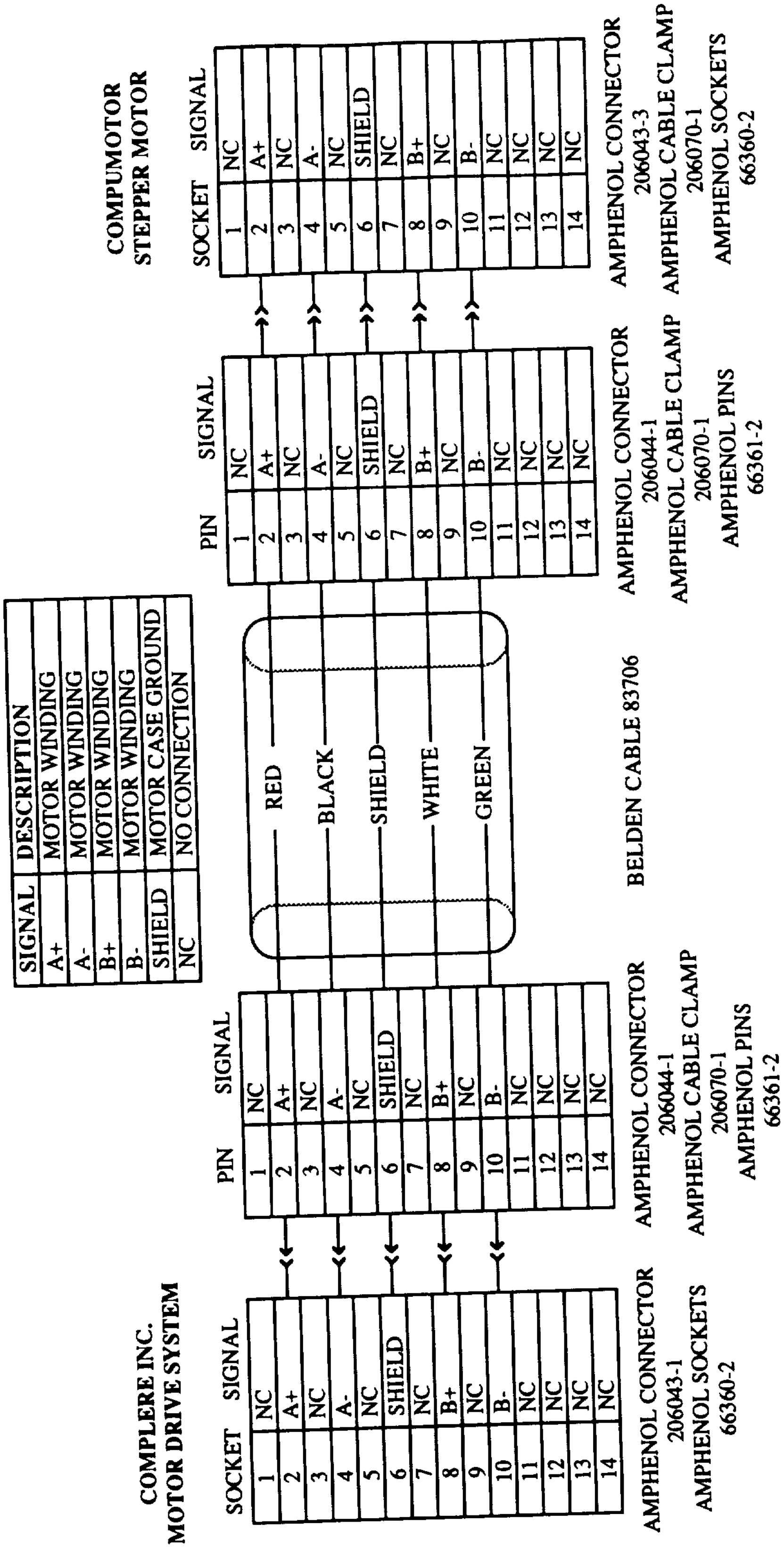


Figure 6 Motor Drive System to Compumotor Stepper Motor Cable.

SIGNAL	DESCRIPTION
A+	QUADRATURE ENCODER SIGNAL
A-	LOGICAL COMPLEMENT OF A+
B+	QUADRATURE ENCODER SIGNAL
B-	LOGICAL COMPLEMENT OF B+
Z+	ONCE PER REVOLUTION
Z-	LOGICAL COMPLEMENT OF Z+
SHIELD	CASE GROUND
GROUND	LOGICAL GROUND

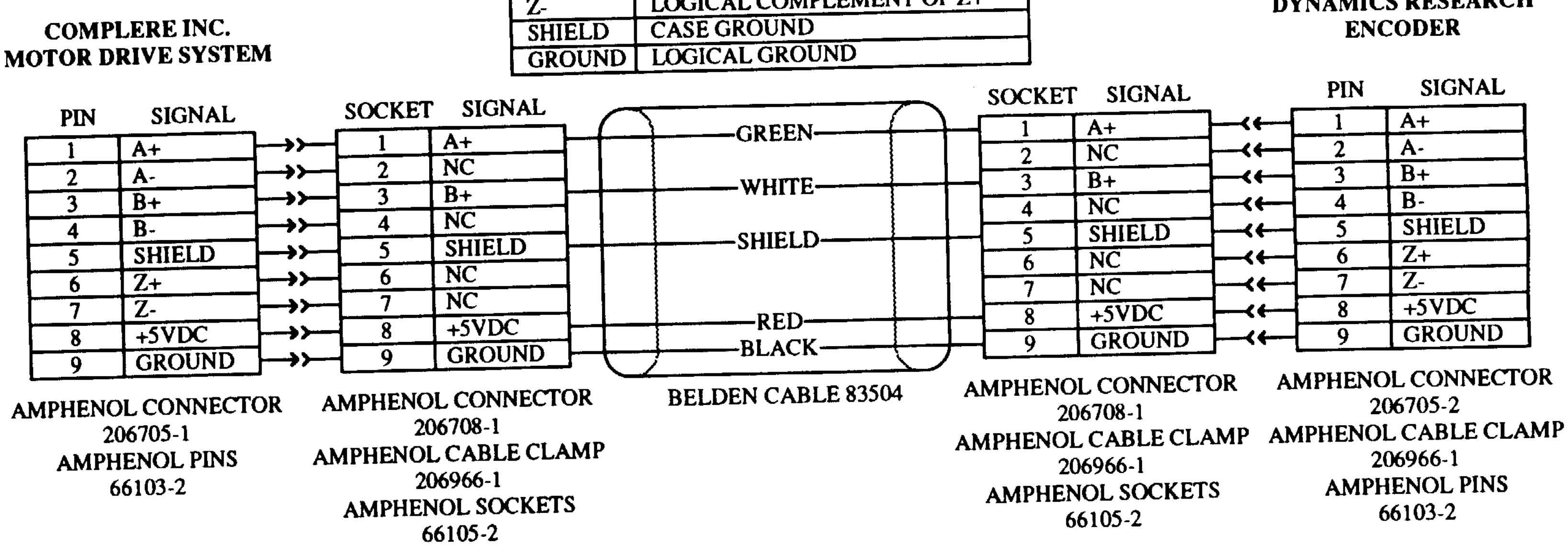


Figure 7 Motor Drive System to Dynamics Research Encoder Cable.

SIGNAL	DESCRIPTION
HOME	HOME SWITCH SIGNAL
CW	END OF TRAVEL LIMIT SIGNAL CLOCKWISE
CCW	END OF TRAVEL LIMIT SIGNAL COUNTER CLOCKWISE
COMMON	LOGICAL GROUND

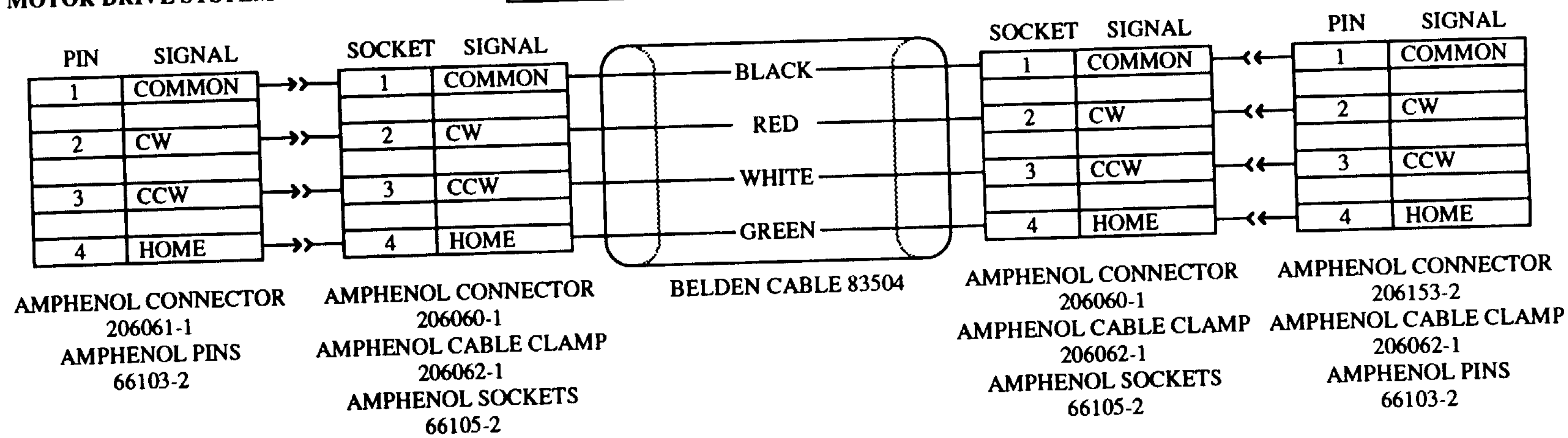


Figure 8 Motor Drive System to Linear Industries Limit Switches Cable.

COMPLERE INC.
MOTOR DRIVE SYSTEM

C1 - CHANNEL 1
C2 - CHANNEL 2
C3 - CHANNEL 3
C4 - CHANNEL 4

TCS8
ENCODER

SOCKET	SIGNAL	PIN	SIGNAL	SOCKET	SIGNAL	PIN	SIGNAL
1	C1 +5V	1	C1 +5V	1	C1 +5V	1	C1 +5V
2	C1 B+	2	C1 B+	2	C1 B+	2	C1 B+
3	C1 SHIELD	3	C1 SHIELD	3	C1 SHIELD	3	C1 SHIELD
4	C2 A+	4	C2 A+	4	C2 A+	4	C2 A+
5	C2 GND	5	C2 GND	5	C2 GND	5	C2 GND
6	C3 +5V	6	C3 +5V	6	C3 +5V	6	C3 +5V
7	C3 B+	7	C3 B+	7	C3 B+	7	C3 B+
8	C3 SHIELD	8	C3 SHIELD	8	C3 SHIELD	8	C3 SHIELD
9	C4 A+	9	C4 A+	9	C4 A+	9	C4 A+
10	C4 GND	10	C4 GND	10	C4 GND	10	C4 GND
11	NC	11	NC	11	NC	11	NC
12	NC	12	NC	12	NC	12	NC
13	NC	13	NC	13	NC	13	NC
14	C3 A+	14	C3 A+	14	C3 A+	14	C3 A+
15	C3 GND	15	C3 GND	15	C3 GND	15	C3 GND
16	C4 +5V	16	C4 +5V	16	C4 +5V	16	C4 +5V
17	C4 B+	17	C4 B+	17	C4 B+	17	C4 B+
18	C4 SHIELD	18	C4 SHIELD	18	C4 SHIELD	18	C4 SHIELD
19	C3 A+	19	C3 A+	19	C3 A+	19	C3 A+
20	C3 GND	20	C3 GND	20	C3 GND	20	C3 GND
21	C4 +5V	21	C4 +5V	21	C4 +5V	21	C4 +5V
22	C4 B+	22	C4 B+	22	C4 B+	22	C4 B+
23	C4 SHIELD	23	C4 SHIELD	23	C4 SHIELD	23	C4 SHIELD
24	NC	24	NC	24	NC	24	NC
25	NC	25	NC	25	NC	25	NC

DB-25 FEMALE — DB-25 MALE — 6 FT. DB-25 STRAIGHT THROUGH EXTENSION CABLE — DB-25 FEMALE — DB-25 MALE

Figure 9 Motor Drive System to TCS8 Encoder Signals Cable.

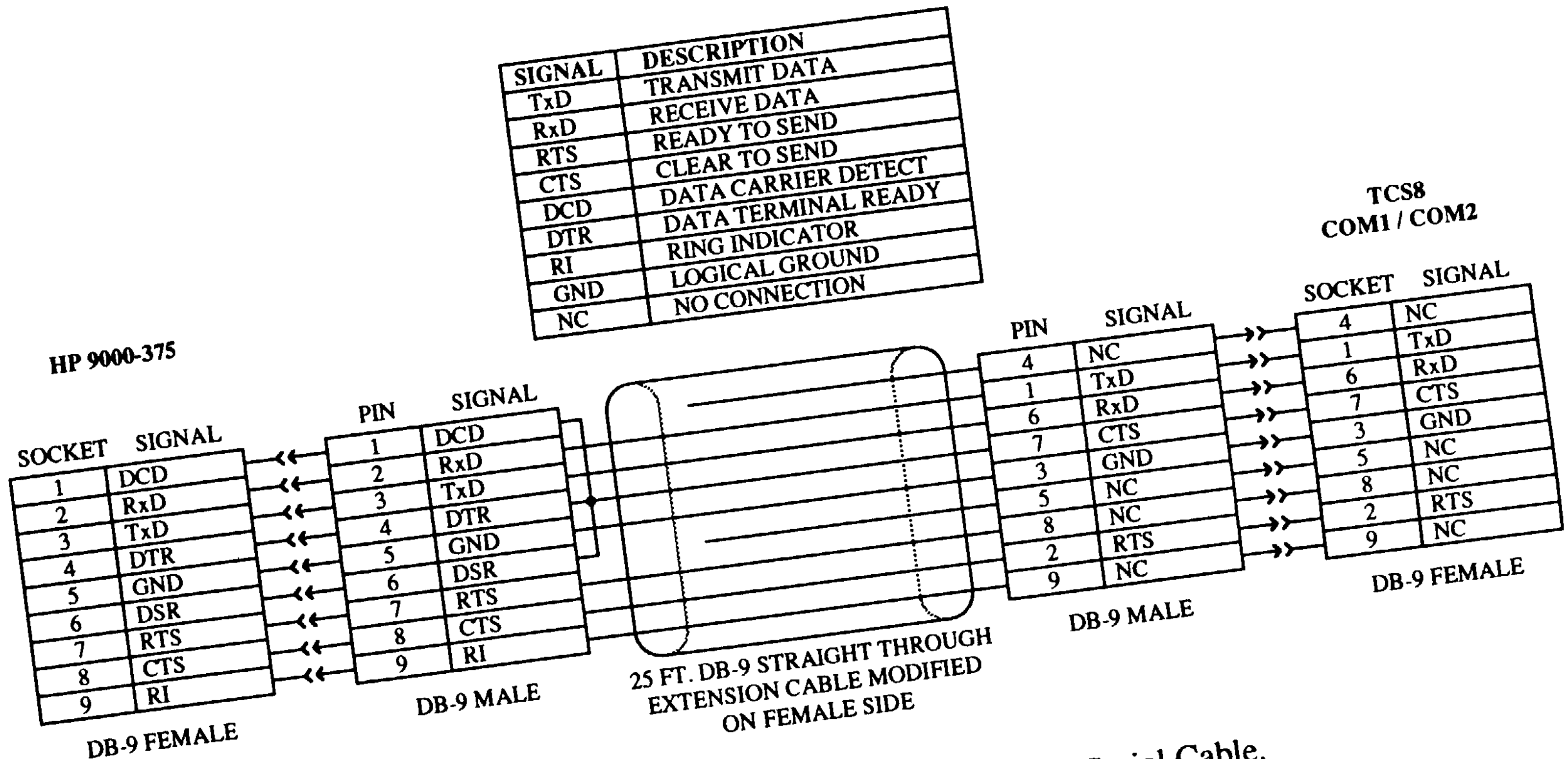


Figure 10 HP Series 9000 Model 375 to TCS8 Serial Cable.

SIGNAL	DESCRIPTION
TxD	TRANSMIT DATA
RxD	RECEIVE DATA
RTS	READY TO SEND
CTS	CLEAR TO SEND
GND	LOGICAL GROUND
NC	NO CONNECTION

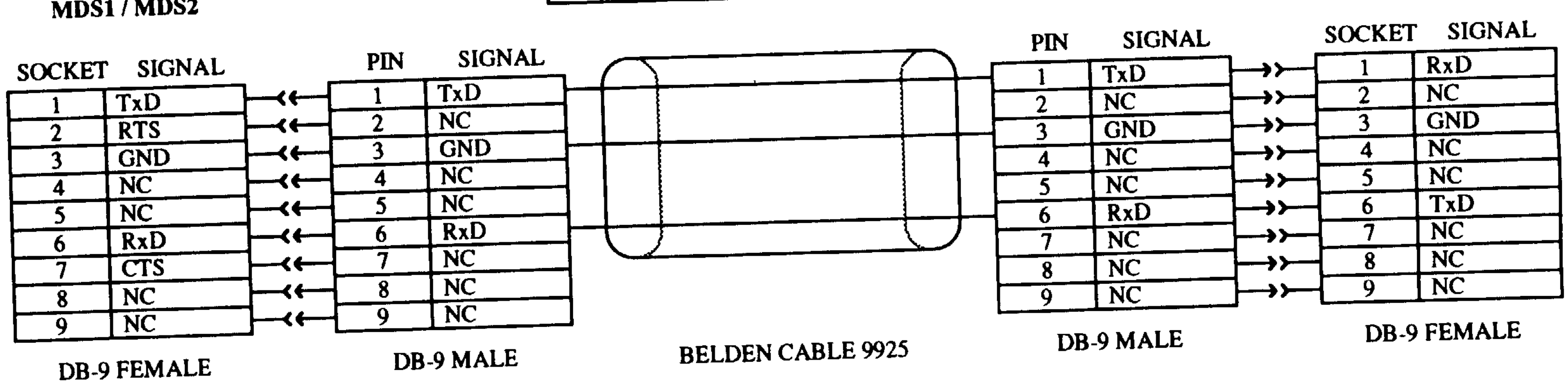


Figure 11 TCS8 to Motor Drive System Serial Cable.

APPENDIX A

ORIGINAL SOFTWARE CODE LISTING.

APPENDIX A

Original Software Code Listing.

TABLE OF CONTENTS

Hard Disk Directory Catalog Listing.

```
:CS80, 1400, 0, 0
VOLUME LABEL: B9826
FILE NAME PRO TYPE  REC/FILE BYTE/REC  ADDRESS     DATE      TIME

SYSB60       SYSTM     3388      256        32   17-Jul-91 13:10
CDUMP6       PROG        44      256      3420   17-Jul-91 13:10
BPLOT6       PROG        40      256      3464   17-Jul-91 13:10
AUTOST       PROG        10      256      3504   17-Jul-91 13:10
ARRAY        BDAT        50      256      3515   17-Jul-91 13:11
KEYS         BDAT         4      256      3566   17-Jul-91 13:11
COPY         PROG        25      256      3570   17-Jul-91 13:11
3.5'HWT91    PROG       372      256      3595   17-Jan-92 16:03
```

```
100 !       ****************************************************
110 !       *                                                  *
120 !       *          Property of COMPLERE INC.               *
130 !       *          Proprietary software                    *
140 !       *          Copyright June 18, 1991                 *
150 !       *          Developed by: T. Kevin McDevitt         *
160 !       *                                                  *
170 !       *          LASER DOPPLER VELOCIMETER TEST          *
180 !       *                                                  *
190 !       *          3.5 FOOT HYPER SONIC WIND TUNNEL        *
200 !       *              NASA AMES RESEARCH CENTER           *
210 !       *                                                  *
220 !       ****************************************************
230                 !
240                 DEG
250                 !
260                 OPTION BASE 1
270                 COM /Data/ INTEGER Raw(1000,10),Valid(1000),REAL Table(0:32766),Ui(1000),Vi(1000),Wi(1000),Ai(1000),
                       Bi(1000),Ii(1000),Ci(1000)
280                 COM /Array/ Name$(100,4)[10],Image$(100,4)[10],Units$(100,4)[10],REAL Array(100,4)
290                 COM /Pos/ Pname$(25,1)[10],Pimage$(25,1)[10],Punits$(25,1)[10],REAL Pos(25,1),Npos
300                 COM /Graph/ Wndw(9,4),Vwprt(9,4),Xdiv(9),Ydiv(9),Xlabel$(9)[80],Ylabel$(9)[80],Title$(9)[80],
                       Ximage$(9)[80],Yimage$(9)[80],Legend$(9,5)[80]
310                 COM Run,File,Paxis
320                 !
330                 DIM Menu$(5,8)[80],System$[20],Data$[20],File$[50],L$[160]
340                 INTEGER Gsave(1280,1024),At_exp,Ct_exp,Cmask,Nsam,N(10,3)
350                 REAL Atime,Ctime,Sum(10,3),Symbols(5,0:20,3)
360                 !
370                 DIM Tcs2tun1(4,4),Tun2tcs1(4,4),Tun2mod(3,3),Tun2ldv(3,3),Tun(4),Tcs1(4)
380                 DIM Tcs2tun2(4,4),Tun2tcs2(4,4),Mod2tun(3,3),Ldv2tun(3,3),Mod(4),Tcs2(4)
390                 !
400                 DIM Beam_spc(3),Focl_len(3),Mea_sgn(3),Mix_frq(3),Mix_sgn(3),Frng_spc(3),Thata(3,3)
410                 DIM Beam_sep(3),Wave_len(3),Brg_frq(3),Brg_sgn(3),Index(3),Coin(3)
420                 !
430                 PRINTER IS CRT
440                 CLEAR SCREEN
450                 GCLEAR
460                 !
470                 GOSUB Lvds_set_up
480                 GOSUB File_set_up
490                 GOSUB Tcs8_set_up
500                 GOSUB Menu_set_up
510                 GOSUB Grph_set_up
520                 GOTO 580
530                 CLEAR SCREEN
540                 CALL Tcs8move(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Mod2tun(*),Tun2tcs1(*),Tun2tcs2(*),"Tx & Rx","LASER",
                       "ABSOLUTE",2,.250)
550                 CALL Tcs8move(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Mod2tun(*),Tun2tcs1(*),Tun2tcs2(*),"Tx & Rx","LASER",
                       "ABSOLUTE",2,.251)
560                 GOTO 540
570                 !
580                 Date=TIMEDATE
590                 Time=Date
600                 CALL Purge(System$,Data$)
610 Here:           DISP TIME$(TIMEDATE),DATE$(TIMEDATE)
620                 GOTO Here
630                 STOP
640 File_set_up:    System$=":,1400,0,0"
650                 Data$=":,1400,0,1"
660                 LOAD KEY "KEYS"&System$
670                 IF NOT INMEM("Gdump_colored") THEN LOADSUB ALL FROM "CDUMP6"&System$
680                 IF NOT INMEM("Bload") THEN LOADSUB ALL FROM "BPLOT6"&System$
690                 IF NOT INMEM("Bstore") THEN LOADSUB ALL FROM "BPLOT6"&System$
700                 GOSUB Read_array
710                 GOSUB Read_calc_fill
720                 GOSUB Save_array
730                 CLEAR SCREEN
740                 RETURN
750 Grph_set_up:    CALL Read_symbols(Symbols(*))
760                 CALL Crt_init
770                 CALL Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
780                 RETURN
790 Menu_set_up:    CALL Menu_read(Menu$(*))
800                 CALL Menu_disp(Menu,Menu$(*))
810                 GOSUB On_key
820                 Busy=0
830                 Ready=1
840                 RETURN
850 Tcs8_set_up:    CALL Tcs8init(@Tcs8)
```

```
860                   CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
870                   GOSUB Calc
880                   GOSUB Fill
890                   RETURN
900 Lvds_set_up:      CALL Lvdas_init(@Lvdas)
910                   CALL Table(Table(*))
920                   RETURN
930 On_key:           ON KEY 1 GOSUB Key1
940                   ON KEY 2 GOSUB Key2
950                   ON KEY 3 GOSUB Key3
960                   ON KEY 4 GOSUB Key4
970                   ON KEY 5 GOSUB Key5
980                   ON KEY 6 GOSUB Key6
990                   ON KEY 7 GOSUB Key7
1000                  ON KEY 8 GOSUB Key8
1010                  RETURN
1020 Key1:            Key=1
1030                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1040                  ON Menu GOSUB M1k1,M2k1,M3k1,M4k1,M5k1,M6k1,M7k1
1050                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1060                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1070                  RETURN
1080 Key2:            Key=2
1090                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1100                  ON Menu GOSUB M1k2,M2k2,M3k2,M4k2,M5k2,M6k2,M7k2
1110                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1120                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1130                  RETURN
1140 Key3:            Key=3
1150                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1160                  ON Menu GOSUB M1k3,M2k3,M3k3,M4k3,M5k3,M6k3,M7k3
1170                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1180                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1190                  RETURN
1200 Key4:            Key=4
1210                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1220                  ON Menu GOSUB M1k4,M2k4,M3k4,M4k4,M5k4,M6k4,M7k4
1230                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1240                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1250                  RETURN
1260 Key5:            Key=5
1270                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1280                  ON Menu GOSUB M1k5,M2k5,M3k5,M4k5,M5k5,M6k5,M7k5
1290                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1300                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1310                  RETURN
1320 Key6:            Key=6
1330                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1340                  ON Menu GOSUB M1k6,M2k6,M3k6,M4k6,M5k6,M6k6,M7k6
1350                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1360                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1370                  RETURN
1380 Key7:            Key=7
1390                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1400                  ON Menu GOSUB M1k7,M2k7,M3k7,M4k7,M5k7,M6k7,M7k7
1410                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1420                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1430                  RETURN
1440 Key8:            Key=8
1450                  CALL Menu_status(Menu,Key,Busy,Menu$(*))
1460                  ON Menu GOSUB M1k8,M2k8,M3k8,M4k8,M5k8,M6k8,M7k8
1470                  CALL Menu_status(Menu,Key,Ready,Menu$(*))
1480                  CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1490                  RETURN
1500 M1k1:            Menu=2
1510                  CALL Menu_disp(Menu,Menu$(*))
1520                  RETURN
1530 M1k2:            Menu=3
1540                  CALL Menu_disp(Menu,Menu$(*))
1550                  RETURN
1560 M1k3:            KEY LABELS OFF
1570                  PRINTER IS CRT;WIDTH 132
1580                  DISP ""
1590                  FOR L=1 TO 9
1600                      PRINT TABXY(1,L);RPT$(" ",120)
1610                  NEXT L
1620                  PRINTER IS PRT
1630                  PRINT USING "#,@"
1640                  DUMP GRAPHICS
1650                  PRINT USING "#,@"
```

```
1660              PRINTER IS CRT
1670              CALL Menu_disp(Menu,Menu$(*))
1680              RETURN
1690 M1k4:        CALL Enter_value("number of traverse positions",Npos,"K")
1700              REDIM Pos(Npos,1),Pname$(Npos,1),Pimage$(Npos,1),Punits$(Npos,1)
1710              MAT Pimage$= ("M4D.4D")
1720              MAT Punits$= ("in")
1730              FOR K=1 TO Npos
1740                 Pname$(K,1)="Pos#"&VAL$(K)
1750              NEXT K
1760              GSTORE Gsave(*)
1770              CALL Change("VALUES",Pos(*),Pname$(*),Pimage$(*),Punits$(*))
1780              GLOAD Gsave(*)
1790              CALL Menu_disp(Menu,Menu$(*))
1800              RETURN
1810 M1k5:        GOSUB Read_calc_fill
1820              CALL Enter_value(CHR$(NUM("X")+Paxis-1),Mod(Paxis),"K")
1830 M1k5a:       ON KBD CALL Do_nothing
1840              DISP "Moving"
1850              Movement=Mod(Paxis)
1860              CALL Tcs8move(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Mod2tun(*),Tun2tcs1(*),Tun2tcs2(*),"Tx &
                     Rx","MODEL","ABSOLUTE",Paxis,Movement)
1870              CALL Tcs8print(Mod(*),Tun(*),Tcs1(*),Tcs2(*))
1880              CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
1890              GOSUB Calc
1900              GOSUB Fill
1910              DISP ""
1920              OFF KBD
1930              RETURN
1940 M1k6:        CALL Fix(Array(*),Name$(*),Image$(*),Units$(*))
1950              DISP "Press any key to TAKE DATA"
1960              CALL Rt_histo(@Lvdas,Symbols(*),1)
1970              Cmask=Coin(1)*1+Coin(2)*2+Coin(3)*4
1980              Nsam=MIN(Nreads,1000)
1990              Date=TIMEDATE
2000              Time=Date
2010              Atime=10
2020              CALL Lvdas_take(@Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Nsam)
2030              IF Nsam>1 THEN
2040                 OUTPUT PRT;RPT$("=",140)
2050                 CALL Data_reduce(At_exp,Ct_exp,Nsam)
2060                !CALL Data_histo(Array(*),Nsam)
2070                 CALL Pt_histo(Symbols(*),Run,File,Mod(Paxis),Nsam)
2080                 CALL Data_clip(Nsam,Umin,Umax,Vmin,Vmax)
2090                 CALL Data_sum(Sum(*),N(*),Nsam)
2100                 CALL Data_calc(N(*),Sum(*),U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
2110                 CALL
                     Data_print(Paxis,Mod(Paxis),Nsam,"MHz",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                     a1,W1a1)
2120                 CALL Data_fconvert(Array(*))
2130                 CALL Data_aconvert(Gain)
2140                 CALL Data_sum(Sum(*),N(*),Nsam)
2150                 CALL Data_calc(N(*),Sum(*),U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
2160                 CALL
                     Data_print(Paxis,Mod(Paxis),Nsam,"LDV",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                     a1,W1a1)
2170                 CALL Data_trnsfrm(Ldv2tun(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1)
2180                 CALL
                     Data_print(Paxis,Mod(Paxis),Nsam,"TUN",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                     a1,W1a1)
2190                 CALL Data_trnsfrm(Tun2mod(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1)
2200                 CALL
                     Data_print(Paxis,Mod(Paxis),Nsam,"MOD",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                     a1,W1a1)
2210                 CALL Data_plot(Array(*),Symbols(*),6,Mod(Paxis),U,V,W,1/Uinf,N(1,1),N(2,1),N(3,1))
2220                 CALL Data_plot(Array(*),Symbols(*),7,Mod(Paxis),U1,V1,W1,1/Uinf,N(1,2),N(2,2),N(3,2))
2230                 CALL Data_plot(Array(*),Symbols(*),8,Mod(Paxis),U1v1,V1w1,W1u1,1/Uinf^2,N(1,3),N(2,3),N(3,3))
2240                 CALL Data_plot(Array(*),Symbols(*),9,Mod(Paxis),Ttemp,Uinf,Uedge,1,N(4,1),1,1)
2250                 OUTPUT PRT;RPT$("=",140)
2260                 GOSUB Store_file
2270                 File=File+1
2280              END IF
2290              RETURN
2300 M1k7:        Quit=0
2310              ON KBD GOSUB Quit
2320              FOR J=1 TO Npos
2330                 Mod(Paxis)=Pos(J,1)
2340                 GOSUB M1k5a
2350                 GOSUB M1k6
2360                 IF Quit THEN 2380
```

```
2370           NEXT J
2380           OFF KBD
2390           GOSUB On_key
2400           CALL Menu_disp(Menu,Menu$(*))
2410           RETURN
2420 M1k8:     DISP "Press any key to return to main menu"
2430           CALL Rt_histo(@Lvdas,Symbols(*),1)
2440           RETURN
2450 M2k1:     Menu=1
2460           CALL Menu_disp(Menu,Menu$(*))
2470           RETURN
2480 M2k2:     SELECT TRIM$(Menu$(Menu,Key)[20])
2490           CASE "Tx & Rx"
2500             Menu$(Menu,Key)[20]="Tx"
2510           CASE "Tx"
2520             Menu$(Menu,Key)[20]="Rx"
2530           CASE "Rx"
2540             Menu$(Menu,Key)[20]="Tx & Rx"
2550           END SELECT
2560           CALL Menu_disp(Menu,Menu$(*))
2570           RETURN
2580 M2k3:     SELECT TRIM$(Menu$(Menu,Key)[20])
2590           CASE "MODEL"
2600             Menu$(Menu,Key)[20]="TUNNEL"
2610           CASE "TUNNEL"
2620             Menu$(Menu,Key)[20]="LASER"
2630           CASE "LASER"
2640             Menu$(Menu,Key)[20]="MODEL"
2650           END SELECT
2660           CALL Menu_disp(Menu,Menu$(*))
2670           RETURN
2680 M2k4:     SELECT TRIM$(Menu$(Menu,Key)[20])
2690           CASE "ABSOLUTE"
2700             Menu$(Menu,Key)[20]="RELATIVE"
2710           CASE "RELATIVE"
2720             Menu$(Menu,Key)[20]="ABSOLUTE"
2730           END SELECT
2740           CALL Menu_disp(Menu,Menu$(*))
2750           RETURN
2760 M2k5:     !
2770 M2k6:     !
2780 M2k7:     !
2790 M2k8:     Side$=TRIM$(Menu$(Menu,2)[20])
2800           Coor$=TRIM$(Menu$(Menu,3)[20])
2810           Mode$=TRIM$(Menu$(Menu,4)[20])
2820           CALL Enter_value(Mode$&" Movement",Movement,"4D.5D")
2830           ON KBD CALL Do_nothing
2840           DISP "Moving"
2850           CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
2860           CALL Tcs8move(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Mod2tun(*),Tun2tcs1(*),Tun2tcs2(*),Side$,Coor$,Mode$,Key-
                 4,Movement)
2870           CALL Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
2880           DISP ""
2890           OFF KBD
2900           RETURN
2910 M3k1:     Menu=1
2920           CALL Menu_disp(Menu,Menu$(*))
2930           RETURN
2940 M3k2:     CALL Enter_value("Run",Run,"3D.2D")
2950           CALL Enter_value("File",File,"3D")
2960           RETURN
2970 M3k3:     CALL Enter_value("Number of Samples ",Nreads,"K")
2980           RETURN
2990 M3k4:     CALL Enter_string("Traverse Axis for Profile ",Paxis$,"K")
3000           SELECT Paxis$
3010           CASE "X"
3020             Paxis=1
3030           CASE "Y"
3040             Paxis=2
3050           CASE "Z"
3060             Paxis=3
3070           CASE "A"
3080             Paxis=4
3090           CASE ELSE
3100             GOTO M3k4
3110           END SELECT
3120           RETURN
3130 M3k5:     GOSUB Read_calc_fill
3140 M3k5a:   !OUTPUT PRT USING "#,@,2/"
3150           OUTPUT PRT USING "#,2/"
```

```
3160          OUTPUT PRT USING "20X,K,/";"TRAVERSE COORDINATE TRANSFORMATION MATRICIES"
3170          OUTPUT PRT USING "20X,K,/,4(13X,4(8D.5D),/)";"Transmitting side TCS8 to TUNNEL",Tcs2tunl(*)
3180          OUTPUT PRT USING "20X,K,/,4(13X,4(8D.5D),/)";"Receiving side TCS8 to TUNNEL",Tcs2tun2(*)
3190          OUTPUT PRT USING "20X,K,/,4(13X,4(8D.5D),/)";"Transmitting side TUNNEL to TCS8",Tun2tcsl(*)
3200          OUTPUT PRT USING "20X,K,/,4(13X,4(8D.5D),/)";"Receiving side TUNNEL to TCS8",Tun2tcs2(*)
3210          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"TUNNEL to MODEL",Tun2mod(*)
3220          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"MODEL to TUNNEL",Mod2tun(*)
3230          OUTPUT PRT USING "20X,K,/";"VELOCITY COORDINATE TRANSFORMATION MATRICIES"
3240          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"LASER to TUNNEL",Ldv2tun(*)
3250          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"TUNNEL to LASER",Tun2ldv(*)
3260          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"TUNNEL to MODEL",Tun2mod(*)
3270          OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"MODEL to TUNNEL",Mod2tun(*)
3280          OUTPUT PRT USING "#,@"
3290          RETURN
3300 M3k6:    CALL Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
3310          RETURN
3320 M3k7:    Menu=4
3330          CALL Menu_disp(Menu,Menu$(*))
3340          RETURN
3350 M3k8:    Menu=5
3360          CALL Menu_disp(Menu,Menu$(*))
3370          RETURN
3380 M4k1:    Menu=3
3390          CALL Menu_disp(Menu,Menu$(*))
3400          RETURN
3410 M4k2:    GOSUB Read_array
3420          GOSUB Read_calc_fill
3430          RETURN
3440 M4k3:    GOSUB Read_calc_fill
3450          GOSUB Save_array
3460          RETURN
3470 M4k4:    GOSUB Read_calc_fill
3480          GOSUB Print_header
3490          RETURN
3500 M4k5:    GSTORE Gsave(*)
3510          GOSUB Read_calc_fill
3520          CALL Change("VALUES",Array(*),Name$(*),Image$(*),Units$(*))
3530          GOSUB Read_calc_fill
3540          GLOAD Gsave(*)
3550          RETURN
3560 M4k6:    GSTORE Gsave(*)
3570          GOSUB Read_calc_fill
3580          CALL Change("NAMES",Array(*),Name$(*),Image$(*),Units$(*))
3590          GOSUB Read_calc_fill
3600          GLOAD Gsave(*)
3610          RETURN
3620 M4k7:    GSTORE Gsave(*)
3630          GOSUB Read_calc_fill
3640          CALL Change("UNITS",Array(*),Name$(*),Image$(*),Units$(*))
3650          GOSUB Read_calc_fill
3660          GLOAD Gsave(*)
3670          RETURN
3680 M4k8:    GSTORE Gsave(*)
3690          GOSUB Read_calc_fill
3700          CALL Change("IMAGES",Array(*),Name$(*),Image$(*),Units$(*))
3710          GOSUB Read_calc_fill
3720          GLOAD Gsave(*)
3730          RETURN
3740 M5k1:    Menu=3
3750          CALL Menu_disp(Menu,Menu$(*))
3760          RETURN
3770 M5k2:    CALL Tcs8set("P",@Tcs8)          ! View and set TCS8 Positions
3780          GRAPHICS ON
3790          CALL Menu_disp(Menu,Menu$(*))
3800          RETURN
3810 M5k3:    CALL Tcs8set("U",@Tcs8)          ! View and set TCS8 counts per Unit length
3820          GRAPHICS ON
3830          CALL Menu_disp(Menu,Menu$(*))
3840          RETURN
3850 M5k4:    CALL Tcs8set("R",@Tcs8)          ! View and set TCS8 counts per Revolution
3860          GRAPHICS ON
3870          CALL Menu_disp(Menu,Menu$(*))
3880          RETURN
3890 M5k5:    CALL Tcs8set("V",@Tcs8)          ! View and set TCS8 Velocities
3900          GRAPHICS ON
3910          CALL Menu_disp(Menu,Menu$(*))
3920          RETURN
3930 M5k6:    CALL Tcs8set("A",@Tcs8)          ! View and set TCS8 Accelerations
3940          GRAPHICS ON
3950          CALL Menu_disp(Menu,Menu$(*))
```

```
3960          RETURN
3970 M5k7:    CALL Enter_value("Run",Run,"3D.2D")
3980          CALL Enter_value("File",File,"3D")
3990          FOR Run=Run TO Run
4000              CLEAR SCREEN
4010              FOR File=1 TO 100
4020                  GOSUB Read_file
4030                  IF File$="" THEN 4170
4040                  CALL Data_reduce(At_exp,Ct_exp,Nsam)
4050                  Vwprt(1,1)=50
4060                  Vwprt(1,2)=225
4070                  Vwprt(2,1)=275
4080                  Vwprt(2,2)=450
4090                  Vwprt(4,1)=500
4100                  Vwprt(4,2)=675
4110                  FOR G=1 TO 5
4120                      Vwprt(G,3)=1025-65*File
4130                      Vwprt(G,4)=1065-65*File
4140                  NEXT G
4150                  CALL Pt_histo(Symbols(*),Run,File,Mod(Paxis),Nsam)
4160              NEXT File
4170              DISP ""
4180              PRINTER IS PRT
4190              PRINT USING "#,@"
4200              DUMP GRAPHICS
4210              PRINT USING "#,@"
4220              PRINTER IS CRT
4230          NEXT Run
4240          CLEAR SCREEN
4250          CALL Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
4260          RETURN
4270 M5k8:    CALL Enter_value("Run",Run,"3D.2D")
4280          CALL Enter_value("File",File,"3D")
4290          GOSUB Read_file
4300          IF File$="" THEN RETURN
4310          GOSUB Print_header
4320          CALL Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
4330         !BEEP
4340         !DISP "SWITCH SWITCH TO B AND THEN PRESS <CONTINUE>"
4350         !PAUSE
4360          FOR File=1 TO 100
4370              GOSUB Read_file
4380              IF File$="" THEN 4630
4390              Cmask=Coin(1)*1+Coin(2)*2+Coin(3)*4
4400              OUTPUT PRT;RPT$("=",140)
4410              CALL Data_reduce(At_exp,Ct_exp,Nsam)
4420             !CALL Data_xfer(@Tcs8,Run,File,Ui(*),Vi(*),Ai(*),Valid(*),Nsam)
4430              CALL Pt_histo(Symbols(*),Run,File,Mod(Paxis),Nsam)
4440              CALL Data_clip(Nsam,Umin,Umax,Vmin,Vmax)
4450              CALL Data_sum(Sum(*),N(*),Nsam)
4460              CALL Data_calc(N(*),Sum(*),U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
4470              CALL
                  Data_print(Paxis,Mod(Paxis),Nsam,"MHz",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                  a1,W1a1)
4480              CALL Data_fconvert(Array(*))
4490              CALL Data_aconvert(Gain)
4500              CALL Data_sum(Sum(*),N(*),Nsam)
4510              CALL Data_calc(N(*),Sum(*),U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
4520              CALL
                  Data_print(Paxis,Mod(Paxis),Nsam,"LDV",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                  a1,W1a1)
4530              CALL Data_trnsfrm(Ldv2tun(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1)
4540              CALL
                  Data_print(Paxis,Mod(Paxis),Nsam,"TUN",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                  a1,W1a1)
4550              CALL Data_trnsfrm(Tun2mod(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1)
4560              CALL
                  Data_print(Paxis,Mod(Paxis),Nsam,"MOD",U,V,W,A,B,I0,C0,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1
                  a1,W1a1)
4570              CALL Data_plot(Array(*),Symbols(*),6,Mod(Paxis),U,V,W,1/Uinf,N(1,1),N(2,1),N(3,1))
4580              CALL Data_plot(Array(*),Symbols(*),7,Mod(Paxis),U1,V1,W1,1/Uinf,N(1,2),N(2,2),N(3,2))
4590              CALL Data_plot(Array(*),Symbols(*),8,Mod(Paxis),U1v1,V1w1,W1u1,1/Uinf^2,N(1,3),N(2,3),N(3,3))
4600              CALL Data_plot(Array(*),Symbols(*),9,Mod(Paxis),Ttemp,Uinf,Uedge,1,N(4,1),1,1)
4610              OUTPUT PRT;RPT$("=",140)
4620          NEXT File
4630          GOSUB M1k3
4640          File=File-1
4650          GOSUB Read_file
4660          GOSUB Print_header
4670          GOSUB M3k5a
```

```
4680                   PRINTER IS CRT
4690                  !BEEP
4700                  !DISP "SWITCH SWITCH TO A AND THEN PRESS <CONTINUE>"
4710                  !PAUSE
4720                   RETURN
4730 Quit:             Quit=1
4740                   RETURN
4750 Print_header:     PRINTER IS PRT;WIDTH 144
4760                   PRINT USING "#,@,5(K)";CHR$(27)&"&k2S"&CHR$(27)&"&19D"
4770                   CALL Array_print(Array(*),Name$(*),Image$(*),Units$(*))
4780                   PRINT USING "#,@,5(K)";CHR$(27)&"E"
4790                   PRINTER IS CRT
4800                   RETURN
4810 Read_calc_fill:   GOSUB Read
4820                   GOSUB Calc
4830                   GOSUB Fill
4840                   FOR X=1 TO SIZE(Array,2)
4850                       FOR Y=1 TO SIZE(Array,1)
4860                           Array(Y,X)=PROUND(Array(Y,X),-15)
4870                       NEXT Y
4880                   NEXT X
4890                   RETURN
4900 Copy_file:        FOR Run=1.01 TO 6.01
4910                       IF Run=1.01 THEN F2=9
4920                       IF Run=2.01 THEN F2=6
4930                       IF Run=3.01 THEN F2=5
4940                       IF Run=4.01 THEN F2=11
4950                       IF Run=5.01 THEN F2=0
4960                       IF Run=6.01 THEN F2=5
4970                       FOR File=1 TO F2
4980                           Data$=":,1400,0,1"
4990                           GOSUB Read_file
5000                           Data$=":,1400,0,0"
5010                           GOSUB Store_file
5020                           Data$=":,1400,0,1"
5030                       NEXT File
5040                   NEXT Run
5050                   RETURN
5060 Store_header:     DISP "Storing Header"
5070                   File$="R"&VAL$(Run)&Data$
5080                   ON ERROR GOTO 5280
5090                   ASSIGN @Data TO File$
5100                   OFF ERROR
5110                   FOR K=1 TO 10
5120                       WAIT .2
5130                       BEEP
5140                   NEXT K
5150                   CALL Enter_string("Over Write old file ",L$,"K")
5160                   SELECT L$[1,1]
5170                   CASE "Y","y"
5180                       ASSIGN @Data TO *
5190                       PURGE File$
5200                       GOTO 5280
5210                   CASE "N","n"
5220                       CALL Enter_value("Run",Run,"3D.2D")
5230                       CALL Enter_value("File",File,"3D")
5240                       GOTO 5060
5250                   CASE ELSE
5260                       GOTO 5060
5270                   END SELECT
5280                   OFF ERROR
5290                   Fsize=INT((3200+4000*3+128*4+72*4)/256*1.05+1)
5300                   CREATE BDAT File$,Fsize
5310                   ASSIGN @Data TO File$
5320                   OUTPUT @Data;Array(*),Name$(*),Image$(*),Units$(*)
5330                   OUTPUT @Data;Tun2tcs1(*),Tun2tcs2(*),Mod2tun(*),Tun2ldv(*)
5340                   OUTPUT @Data;Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*),Ldv2tun(*)
5350                   ASSIGN @Data TO *
5360                   PROTECT "R"&VAL$(Run)&Data$,"TKM"
5370                   RETURN
5380 Store_file:       GOSUB Calc
5390                   GOSUB Fill
5400                   IF File=1 THEN GOSUB Store_header
5410                   DISP "Storing Data"
5420                   File$="R"&VAL$(Run)&"F"&VAL$(File)&Data$
5430                   ON ERROR GOTO 5630
5440                   ASSIGN @Data TO File$
5450                   OFF ERROR
5460                   FOR K=1 TO 10
5470                       WAIT .2
```

```
5480                    BEEP
5490                NEXT K
5500                CALL Enter_string("Over Write old file ",L$,"K")
5510                SELECT L$[1,1]
5520                CASE "Y","y"
5530                    ASSIGN @Data TO *
5540                    PURGE File$
5550                    GOTO 5630
5560                CASE "N","n"
5570                    CALL Enter_value("Run",Run,"3D.2D")
5580                    CALL Enter_value("File",File,"3D")
5590                    GOTO 5380
5600                CASE ELSE
5610                    GOTO 5380
5620                END SELECT
5630                OFF ERROR
5640                Fsize=INT((3200+Nsam*10*2+60+240)/256*1.05+1)
5650                CREATE BDAT File$,Fsize
5660                ASSIGN @Data TO File$
5670                OUTPUT @Data;Array(*),Raw(*),N(*),Sum(*)
5680                ASSIGN @Data TO *
5690                PROTECT "R"&VAL$(Run)&"F"&VAL$(File)&Data$,"TKM"
5700                RETURN
5710 Read_header:   DISP "Reading Header"
5720                File$="R"&VAL$(Run)&"<TKM>"&Data$
5730                ON ERROR GOTO 5820
5740                ASSIGN @Data TO File$
5750                ENTER @Data;Array(*),Name$(*),Image$(*),Units$(*)
5760                CALL Fix(Array(*),Name$(*),Image$(*),Units$(*))
5770                ENTER @Data;Tun2tcs1(*),Tun2tcs2(*),Mod2tun(*),Tun2ldv(*)
5780                ENTER @Data;Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*),Ldv2tun(*)
5790                ASSIGN @Data TO *
5800                OFF ERROR
5810                RETURN
5820                OFF ERROR
5830                File$=""
5840                RETURN
5850 Read_file:     IF File=1 THEN GOSUB Read_header
5860                DISP "Reading Data"
5870                File$="R"&VAL$(Run)&"F"&VAL$(File)&"<TKM>"&Data$
5880                ON ERROR GOTO 6020
5890                ASSIGN @Data TO File$
5900                ENTER @Data;Array(*)
5910                CALL Fix(Array(*),Name$(*),Image$(*),Units$(*))
5920                GOSUB Read
5930                REDIM Raw(1:Nsam,1:10)
5940                ENTER @Data;Raw(*),N(*),Sum(*)
5950                ASSIGN @Data TO *
5960                OFF ERROR
5970                Date=Array(1,1)                   ! Date
5980                Time=Array(2,1)                   ! Time
5990                Run=Array(3,1)                    ! Run Number
6000                File=Array(4,1)                   ! File Number
6010                RETURN
6020                OFF ERROR
6030                File$=""
6040                RETURN
6050 Fill:          Array(1,1)=Date                   ! Date
6060                Array(1,2)=Mach                   ! Mach Number
6070                Array(1,3)=Stemp                  ! Stagnation Temperature (°R)
6080                Array(1,4)=Alpha1                 ! Angle of Attack
6090                Array(2,1)=Time                   ! Time
6100                Array(2,2)=Temp                   ! Room Temperature (°F)
6110                Array(2,3)=Ttemp                  ! Total Temperautue (°R)
6120                Array(2,4)=Alpha2                 ! Cone angle
6130                Array(3,1)=Run                    ! Run Number
6140                Array(3,2)=Uedge                  ! Uedge
6150                Array(3,3)=Tt_mv                  ! Total Temperautue (mv)
6160                Array(3,4)=Alpha3                 ! Roll angle
6170                Array(4,1)=File                   ! File Number
6180                Array(4,2)=Uinf                   ! Freestreem Velocity
6190                Array(4,3)=Tt_raw                 ! Total Temperautue (raw voltage w/gain)
6200                Array(4,4)=Theta                  ! Tx Side OffAxis Angle
6210                MAT Array(11:14,1)= Mod           ! Probe volume position in Model coordinates
6220                MAT Array(11:14,2)= Tun           ! Probe volume position in Tunnel coordinates
6230                MAT Array(11:14,3)= Tcs1          ! Tx side traverse position in Tcs8 coordinates
6240                MAT Array(11:14,4)= Tcs2          ! Rx side traverse position in Tcs8 coordinates
6250                MAT Array(21,1:3)= Beam_spc       ! Beam spacing at lens
6260                MAT Array(22,1:3)= Focl_len       ! Focal length
6270                MAT Array(23,1:3)= Beam_sep       ! Beam separation agnle in degrees (full angle)
```

```
6280              MAT Array(24,1:3)= Wave_len          ! Wave length
6290              MAT Array(25,1:3)= Frng_spc          ! Fringe spacing
6300              MAT Array(26,1:3)= Brg_frq           ! Bragg frequency
6310              MAT Array(27,1:3)= Mix_frq           ! Mixing frequency
6320              MAT Array(28,1:3)= Mea_sgn           ! Sign of measured frequency in velocity equation
6330              MAT Array(29,1:3)= Brg_sgn           ! Sign of bragg    frequency in velocity equation
6340              MAT Array(30,1:3)= Mix_sgn           ! Sign of mixing   frequency in velocity equation
6350              MAT Array(31,1:3)= Coin              ! Coincedence criteria
6360              MAT Array(32:34,1:3)= Thata          ! Angles between measured (ABC) & tunnel (UVW) coordinate systems
6370              MAT Array(21:23,4)= Index            ! Index of refraction of for laser light (eg: Nair,Nglass,Nwater)
6380              Array(24,4)=Nreads                   ! Number of desired samples
6390              Array(25,4)=Nsam                     ! Number of acquired samples
6400              Array(26,4)=Atime                    ! Acquisition time
6410              Array(27,4)=Ctime                    ! Coincedence time
6420              Array(28,4)=At_exp                   ! Acquisition time exponent
6430              Array(29,4)=Ct_exp                   ! coincedence time exponent
6440              Array(30,4)=Gain                     ! Tunnel Total Temperature Voltage Gain
6450              Array(31,4)=Paxis                    ! Axis for plots
6460              Array(35,1)=Umin                     ! Frequency minimum for U calculation
6470              Array(35,2)=Vmin                     ! Frequency minimum for V calculation
6480              Array(35,3)=Wmin                     ! Frequency minimum for W calculation
6490              Array(36,1)=Umax                     ! Frequency maximum for U calculation
6500              Array(36,2)=Vmax                     ! Frequency maximum for V calculation
6510              Array(36,3)=Wmax                     ! Frequency maximum for W calculation
6520              Array(36,4)=Clip                     ! Clip
6530              RETURN
6540 Read:        !Date=Array(1,1)                     ! Date
6550              Date=TIMEDATE                        ! Date
6560              Mach=Array(1,2)                      ! Mach Number
6570              Stemp=Array(1,3)                     ! Stagnation Temperature (°R)
6580              Alpha1=Array(1,4)                    ! Angle of Attack
6590              !Time=Array(2,1)                     ! Time
6600              Time=Date                            ! Time
6610              Temp=Array(2,2)                      ! Room Temperature (°F)
6620              Ttemp=Array(2,3)                     ! Total Temperautue (°R)
6630              Alpha2=Array(2,4)                    ! Cone angle
6640              !Run=Array(3,1)                      ! Run Number
6650              Uedge=Array(3,2)                     ! Uedge
6660              Tt_mv=Array(3,3)                     ! Total Temperautue (mv)
6670              Alpha3=Array(3,4)                    ! Roll angle
6680              !File=Array(4,1)                     ! File Number
6690              Uinf=Array(4,2)                      ! Freestreem Velocity
6700              Tt_raw=Array(4,3)                    ! Total Temperautue (raw voltage w/gain)
6710              Theta=Array(4,4)                     ! Tx Side OffAxis Angle
6720              MAT Mod= Array(11:14,1)              ! Probe volume position in Model coordinates
6730              MAT Tun= Array(11:14,2)              ! Probe volume position in Tunnel coordinates
6740              MAT Tcs1= Array(11:14,3)             ! Tx side traverse position in Tcs8 coordinates
6750              MAT Tcs2= Array(11:14,4)             ! Rx side traverse position in Tcs8 coordinates
6760              MAT Beam_spc= Array(21,1:3)          ! Beam spacing at lens
6770              MAT Focl_len= Array(22,1:3)          ! Focal length
6780              MAT Beam_sep= Array(23,1:3)          ! Beam separation agnle in degrees (full angle)
6790              MAT Wave_len= Array(24,1:3)          ! Wave length
6800              MAT Frng_spc= Array(25,1:3)          ! Fringe spacing
6810              MAT Brg_frq= Array(26,1:3)           ! Bragg frequency
6820              MAT Mix_frq= Array(27,1:3)           ! Mixing frequency
6830              MAT Mea_sgn= Array(28,1:3)           ! Sign of measured frequency in velocity equation
6840              MAT Brg_sgn= Array(29,1:3)           ! Sign of bragg    frequency in velocity equation
6850              MAT Mix_sgn= Array(30,1:3)           ! Sign of mixing   frequency in velocity equation
6860              MAT Coin= Array(31,1:3)              ! Coincedence criteria
6870              MAT Thata= Array(32:34,1:3)          ! Angles between measured (ABC) & tunnel (UVW) coordinate systems
6880              MAT Index= Array(21:23,4)            ! Index of refraction of for laser light (eg: Nair,Nglass,Nwater)
6890              Nreads=Array(24,4)                   ! Number of desired samples
6900              Nsam=Array(25,4)                     ! Number of acquired samples
6910              Atime=Array(26,4)                    ! Acquisition time
6920              Ctime=Array(27,4)                    ! Coincedence time
6930              At_exp=Array(28,4)                   ! Acquisition time exponent
6940              Ct_exp=Array(29,4)                   ! coincedence time exponent
6950              Gain=Array(30,4)                     ! Tunnel Total Temperature Voltage Gain
6960              Paxis=Array(31,4)                    ! Axis for plots
6970              Umin=Array(35,1)                     ! Frequency minimum for U calculation
6980              Vmin=Array(35,2)                     ! Frequency minimum for V calculation
6990              Wmin=Array(35,3)                     ! Frequency minimum for W calculation
7000              Umax=Array(36,1)                     ! Frequency maximum for U calculation
7010              Vmax=Array(36,2)                     ! Frequency maximum for V calculation
7020              Wmax=Array(36,3)                     ! Frequency maximum for W calculation
7030              Clip=Array(36,4)                     ! Clip
7040              RETURN
7050 Calc:        FOR K=1 TO 3
7060                  IF I=2 THEN
7070                      Beam1=Theta+ATN(Beam_spc(K)/2/Focl_len(K))
```

```
7080                  Beam2=Theta-ATN(Beam_spc(K)/2/Focl_len(K))
7090              ELSE
7100                  Beam1=0+ATN(Beam_spc(K)/2/Focl_len(K))
7110                  Beam2=0-ATN(Beam_spc(K)/2/Focl_len(K))
7120              END IF
7130              Beam1=ASN(Index(1)/Index(3)*SIN(Beam1))
7140              Beam2=ASN(Index(1)/Index(3)*SIN(Beam2))
7150              Beam_sep(K)=Beam1-Beam2
7160              Frng_spc(K)=Wave_len(K)/(2*SIN(Beam_sep(K)/2))/1000
7170          NEXT K
7180          MAT Array(23,1:3)= Beam_sep        ! Beam separation agnle in degrees (full angle)
7190          MAT Array(25,1:3)= Frng_spc        ! Fringe spacing
7200          CALL Ctm_tcs1(Tcs2tun1(*),Tun2tcs1(*))
7210          CALL Ctm_tcs2(Tcs2tun2(*),Tun2tcs2(*))
7220          CALL Ctm_ldv(Index(*),Thata(*),Tun2ldv(*),Ldv2tun(*))
7230          CALL Ctm_mod(Alpha1,Alpha2,Alpha3,Mod2tun(*),Tun2mod(*))
7240          CALL Lvdas_sample_c(@Lvdas,4,Table(*),Vave,Vsdv,Tave,Tsdv)
7250          Tt_raw=Vave
7260          Tt_mv=Tt_raw/Gain*1000
7270          CALL Temp(Mach,Tt_mv,Stemp,Ttemp)
7280          Uinf=Mach*49.0*SQR(Stemp)*.3048
7290         !Uinf=20.043*Mach*SQR((273+5/9*(Temp-32))/(1+.2*Mach^2))
7300          Uedge=Uinf
7310          Cmask=Coin(1)*1+Coin(2)*2+Coin(3)*4
7320          SELECT Paxis
7330          CASE 1
7340              Paxis$="X"
7350          CASE 2
7360              Paxis$="Y"
7370          CASE 3
7380              Paxis$="Z"
7390          CASE 4
7400              Paxis$="A"
7410          CASE ELSE
7420              Paxis=2
7430              Paxis$="Y"
7440              GOTO M3k4
7450          END SELECT
7460          IF Run=0 OR File=0 THEN
7470              CALL Enter_value("Run Number ",Run,"3D.2D")
7480              CALL Enter_value("File Number ",File,"3D")
7490              GOTO 7460
7500          END IF
7510          RETURN
7520 Read_array:  ON ERROR GOTO 7600
7530          ASSIGN @File TO "ARRAY"&System$
7540          ENTER @File;Array(*),Name$(*),Image$(*),Units$(*)
7550          ENTER @File;Tun2tcs1(*),Tun2tcs2(*),Mod2tun(*),Tun2ldv(*)
7560          ENTER @File;Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*),Ldv2tun(*)
7570          ASSIGN @File TO *
7580          OFF ERROR
7590          RETURN
7600          OFF ERROR
7610          ASSIGN @File TO *
7620          ON ERROR GOTO 7640
7630          PURGE "ARRAY"&System$
7640          OFF ERROR
7650          CALL Array_init(Name$(*),Array(*),Image$(*),Units$(*))
7660          CREATE BDAT "ARRAY"&System$,50
7670          GOSUB Save_array
7680          RETURN
7690 Save_array:  ASSIGN @File TO "ARRAY"&System$
7700          OUTPUT @File;Array(*),Name$(*),Image$(*),Units$(*)
7710          OUTPUT @File;Tun2tcs1(*),Tun2tcs2(*),Mod2tun(*),Tun2ldv(*)
7720          OUTPUT @File;Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*),Ldv2tun(*)
7730          ASSIGN @File TO *
7740          RETURN
7750          END
7760 Do_nothing:  SUB Do_nothing
7770              K$=KBD$
7780          SUBEND
7790 Menu:        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
7800 Menu_read:   SUB Menu_read(Menu$(*))
7810              OPTION BASE 1
7820              DIM L$[80]
7830              FOR Menu=1 TO SIZE(Menu$,1)
7840                  FOR Key=1 TO 8
7850                      Menu$(Menu,Key)="M"&VAL$(Menu)&"K"&VAL$(Key)&":"
7860                  NEXT Key
7870              NEXT Menu
```

```
7880                 ON ERROR GOTO 7950
7890                 WHILE 1=1
7900                     READ L$
7910                     Menu=VAL(L$[2,2])
7920                     Key=VAL(L$[4,4])
7930                     Menu$(Menu,Key)=L$
7940                 END WHILE
7950                 SUBEXIT
7960                 DATA "M1K1: Menu2: Laser Alignment"
7970                 DATA         "M2K1: Return to main menu"
7980                 DATA         "M2K2: Sides       : Tx & Rx"
7990                 DATA         "M2K3: Coordinates: MODEL"
8000                 DATA         "M2K4: Mode        : ABSOLUTE"
8010                 DATA         "M2K5: Move X"
8020                 DATA         "M2K6: Move Y"
8030                 DATA         "M2K7: Move Z"
8040                 DATA         "M2K8: Move A"
8050                 DATA "M1K2: Menu3: Pre Run"
8060                 DATA         "M3K1: Return to MAIN menu"
8070                 DATA         "M3K2: Enter Run & File Numbers"
8080                 DATA         "M3K3: Enter Number of Samples"
8090                 DATA         "M3K4: Select Traverse Axis for Profile"
8100                 DATA         "M3K5: Print Coordinate Transformation Matricies"
8110                 DATA         "M3K6: Setup Graphics"
8120                 DATA         "M3K7: Menu4: Tunnel Conditions"
8130                 DATA                 "M4K1: Return to PRE RUN menu"
8140                 DATA                 "M4K2: Load  Tunnel Conditions"
8150                 DATA                 "M4K3: Save  Tunnel Conditions"
8160                 DATA                 "M4K4: Print Tunnel Conditions"
8170                 DATA                 "M4K5: Enter Tunnel Condition Data"
8180                 DATA                 "M4K6: Enter Tunnel Condition Names"
8190                 DATA                 "M4K7: Enter Tunnel Condition Units"
8200                 DATA                 "M4K8: Enter Tunnel Condition Images"
8210                 DATA         "M3K8: Menu5: Traverse"
8220                 DATA                 "M5K1: Return to TRAVERSE menu"
8230                 DATA                 "M5K2: View & Set TCS8 Positions"
8240                 DATA                 "M5K3: View & Set TCS8 Units"
8250                 DATA                 "M5K4: View & Set TCS8 Revolution"
8260                 DATA                 "M5K5: View & Set TCS8 Velocity"
8270                 DATA                 "M5K6: View & Set TCS8 Acceleration"
8280                 DATA "M1K3: Post Run (Dump Graphics)"
8290                 DATA "M1K4: Set Auto Move Positions"
8300                 DATA "M1K5: Move traverse"
8310                 DATA "M1K6: Take data"
8320                 DATA "M1K7: Auto move and take"
8330                 DATA "M1K8: Display Histograms"
8340                SUBEND
8350 Menu_disp:     SUB Menu_disp(Menu,Menu$(*))
8360                    PRINTER IS CRT
8370                    PRINT CHR$(128);
8380                    IF Menu=0 THEN Menu=1
8390                    FOR Key=1 TO 8
8400                        Menu$(Menu,Key)=Menu$(Menu,Key)&RPT$(" ",50-LEN(Menu$(Menu,Key)))
8410                        PRINT TABXY(1,Key);Menu$(Menu,Key)[3]
8420                    NEXT Key
8430                    PRINT CHR$(128);
8440                SUBEND
8450 Menu_status:   SUB Menu_status(Menu,Key,Pen,Menu$(*))
8460                    PRINTER IS CRT
8470                    PRINT TABXY(1,Key);CHR$(129-Pen);Menu$(Menu,Key)[3];CHR$(128)
8480                    WAIT .1
8490                SUBEND
8500 Enter:         !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
8510 Enter_value:   SUB Enter_value(Name$,Value,Image$)
8520                    IF Name$="Date" OR Name$="Time" THEN SUBEXIT
8530                    DISP CHR$(129);
8540                    DISP USING 8550;Name$
8550                    IMAGE #,"Old ",K,"="
8560                    IF Image$<>"" THEN DISP USING "#,"&Image$;Value
8570                    IF Image$="" THEN DISP USING "#,K";Value
8580                    DISP USING 8590;Name$
8590                    IMAGE #,"     Enter new ",K
8600                    INPUT " ? ",Value
8610                    DISP CHR$(128);
8620                SUBEND
8630 Enter_string:  SUB Enter_string(Name$,Value$,Image$)
8640                    DISP CHR$(129);
8650                    DISP USING 8660;Name$
8660                    IMAGE #,"Old ",K,"="
8670                    DISP USING "#,"&Image$;Value$
```

```
8680                 DISP USING 8690;Name$
8690                 IMAGE #,"     Enter new ",K
8700                 INPUT " ? ",Value$
8710                 DISP CHR$(128);
8720             SUBEND
8730 Array:      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
8740 Array_init: SUB Array_init(Name$(*),Array(*),Image$(*),Units$(*))
8750                 ON ERROR GOTO 8930
8760                 READ Y
8770                 FOR X=1 TO SIZE(Name$,2)
8780                     READ Name$(Y,X),Array(Y,X),Image$(Y,X),Units$(Y,X)
8790                     SELECT Image$(Y,X)
8800                     CASE "0"
8810                         Image$(Y,X)="9D"
8820                     CASE "1" TO "7"
8830                         After=VAL(Image$(Y,X))
8840                         Before=8-After
8850                         Image$(Y,X)=VAL$(Before)&"D."&VAL$(After)&"D"
8860                     CASE "K"
8870                     CASE "N"
8880                     CASE ELSE
8890                         Image$(Y,X)="9D"
8900                     END SELECT
8910                 NEXT X
8920                 GOTO 8760
8930                 SUBEXIT
8940                 !     Y   *********X=1*********  *********X=2*********  *********X=3*********  *********X=4********
8950                 DATA  1,  Date      ,  0,0,"" ,  Mach     ,  7.0,4,"" ,  STemp    ,  0,0,°R ,  Alpha1   ,  0,4,°
8960                 DATA  2,  Time      ,  0,0,"" ,  Temp     , 68.5,4,°F ,  TTemp    ,  0,0,°R ,  Alpha2   ,  0,4,°
8970                 DATA  3,  Run       ,  5,2,"" ,  Uedge    ,  1,4,m/s,   Tt       ,  0,3,mv ,  Alpha3   ,  0,4,°
8980                 DATA  4,  File      ,  0,0,"" ,  Uinf     ,  1,4,m/s,   Tt (raw),   0,3,v  ,  Theta    ,  0,4,°
8990                 !     Y   *********X=1*********  *********X=2*********  *********X=3*********  *********X=4********
9000                 DATA 11,  Xmod      ,  0,4,in ,  Xtun     ,  0,4,in ,   X1tcs    ,  0,4,in ,  X2tcs    ,  0,4,in
9010                 DATA 12,  Ymod      ,  0,4,in ,  Ytun     ,  0,4,in ,   Y1tcs    ,  0,4,in ,  Y2tcs    ,  0,4,in
9020                 DATA 13,  Zmod      ,  0,4,in ,  Ztun     ,  0,4,in ,   Z1tcs    ,  0,4,in ,  Z2tcs    ,  0,4,in
9030                 DATA 14,  Amod      ,  0,4,in ,  Atun     ,  0,4,in ,   A1tcs    ,  0,4,in ,  A2tcs    ,  0,4,in
9040                 !     Y   *********X=1*********  *********X=2*********  *********X=3*********  *********X=4********
9050                 DATA 21,  UBeamSpc,.3125,3,in ,  VBeamSpc,.3438,3,in ,  WBeamSpc,.3125,3,in ,  Index1  ,1.000,3,""
9060                 DATA 22,  UFoclLen,30.00,3,in ,  VFoclLen,30.00,3,in ,  WFoclLen,30.00,3,in ,  Index2  ,1.000,3,""
9070                 DATA 23,  UBeamSep,0.000,3,°  ,  VBeamSep,0.000,3,°  ,  WBeamSep,0.000,3,°  ,  Index3  ,1.000,3,""
9080                 DATA 24,  UWaveLen,514.5,3,nm ,  VWaveLen,488.0,3,nm ,  WWaveLen,476.5,3,nm ,  Nreads  , 1000,0,""
9090                 DATA 25,  UFrngSpc,00.00,3,um ,  VFrngSpc,00.00,3,um ,  WFrngSpc,00.00,3,um ,  Nsam    , 1000,0,""
9100                 DATA 26,  Ubrag    ,40.00,4,MHz,  Vbrag   ,40.00,4,MHz,  Wbrag   ,40.00,4,MHz,  Atime   ,    5,6,s
9110                 DATA 27,  Umix     , 0.00,4,MHz,  Vmix    , 0.00,4,MHz,  WMix    , 0.00,4,MHz,  Ctime   , 1E-2,6,s
9120                 DATA 28,  UmeaSgn ,   -1,0,"" ,  VmeaSgn ,   +1,0,"" ,  WmeaSgn ,   +1,0,"" ,  ATexp   ,   12,0,""
9130                 DATA 29,  UbrgSgn ,   +1,0,"" ,  VbrgSgn ,   -1,0,"" ,  WbrgSgn ,   -1,0,"" ,  CTexp   ,    7,0,""
9140                 DATA 30,  UmixSgn ,   -1,0,"" ,  VmixSgn ,   +1,0,"" ,  WmixSgn ,   +1,0,"" ,  Tt Gain ,  100,0,""
9150                 DATA 31,  U coin  ,    1,0,"" ,  V coin  ,    1,0,"" ,  W coin  ,    0,0,"" ,  Paxis   ,    2,0,""
9160                 DATA 32,  ThetaAU ,    0,4,°  ,  ThetaAV ,   90,4,°  ,  ThetaAW ,   90,4,°  ,  ""      ,    0,0,""
9170                 DATA 33,  ThetaBU ,   90,4,°  ,  ThetaBV ,    0,4,°  ,  ThetaBW ,   90,4,°  ,  ""      ,    0,0,""
9180                 DATA 34,  ThetaCU ,   90,4,°  ,  ThetaCV ,   90,4,°  ,  ThetaCW ,    0,4,°  ,  Nose    ,  139,1,cm
9190                 DATA 35,  UFreqMin,    8,4,MHz,  VFreqMin,   25,4,MHz,  WFreqMin,   10,4,MHz,  ""      ,    0,0,""
9200                 DATA 36,  UFreqMax,   32,4,MHz,  VFreqMax,   55,4,MHz,  WFreqMax,   70,4,MHz,  Clip    ,    1,0,""
9210                 !     Y   *********X=1*********  *********X=2*********  *********X=3*********  *********X=4********
9220                 DATA 41,  Xmin1   , 0.00,0,"" ,  Xmax1   ,  100,0,"" ,  Ymin1   ,    0,0,"" ,  Ymax1   ,  100,0,""
9230                 DATA 42,  Xmin2   , 0.00,0,"" ,  Xmax2   ,  100,0,"" ,  Ymin2   ,    0,0,"" ,  Ymax2   ,  100,0,""
9240                 DATA 43,  Xmin3   , 0.00,0,"" ,  Xmax3   ,  100,0,"" ,  Ymin3   ,    0,0,"" ,  Ymax3   ,  100,0,""
9250                 DATA 44,  Xmin4   ,   -1,2,"" ,  Xmax4   ,    1,2,"" ,  Ymin4   ,    0,0,"" ,  Ymax4   ,  100,0,""
9260                 DATA 45,  Xmin5   ,   -1,2,"" ,  Xmax5   ,    1,2,"" ,  Ymin5   ,    0,0,"" ,  Ymax5   ,  100,0,""
9270                 DATA 46,  Xmin6   , -0.5,1,"" ,  Xmax6   ,  1.5,1,"" ,  Ymin6   ,    0,2,"" ,  Ymax6   ,    4,2,""
9280                 DATA 47,  Xmin7   ,    0,1,"" ,  Xmax7   ,   .5,1,"" ,  Ymin7   ,    0,2,"" ,  Ymax7   ,    4,2,""
9290                 DATA 48,  Xmin8   ,   -1,1,"" ,  Xmax8   ,    1,1,"" ,  Ymin8   ,    0,2,"" ,  Ymax8   ,    4,2,""
9300                 DATA 49,  Xmin9   ,    0,0,"" ,  Xmax9   , 2000,0,"" ,  Ymin9   ,    0,2,"" ,  Ymax9   ,    4,2,""
9310                 DATA 51,  Xmin1   ,  935,0,pxl,  Xmax1   , 1235,0,pxl,  Ymin1   ,  725,0,pxl,  Ymax1   ,  825,0,pxl
9320                 DATA 52,  Xmin2   ,  935,0,pxl,  Xmax2   , 1235,0,pxl,  Ymin2   ,  585,0,pxl,  Ymax2   ,  685,0,pxl
9330                 DATA 53,  Xmin3   ,  935,0,pxl,  Xmax3   , 1235,0,pxl,  Ymin3   ,  445,0,pxl,  Ymax3   ,  545,0,pxl
9340                 DATA 54,  Xmin4   ,  935,0,pxl,  Xmax4   , 1235,0,pxl,  Ymin4   ,  305,0,pxl,  Ymax4   ,  405,0,pxl
9350                 DATA 55,  Xmin5   ,  935,0,pxl,  Xmax5   , 1235,0,pxl,  Ymin5   ,  165,0,pxl,  Ymax5   ,  265,0,pxl
9360                 DATA 56,  Xmin6   ,   75,0,pxl,  Xmax6   ,  325,0,pxl,  Ymin6   ,  525,0,pxl,  Ymax6   ,  825,0,pxl
9370                 DATA 57,  Xmin7   ,  425,0,pxl,  Xmax7   ,  675,0,pxl,  Ymin7   ,  525,0,pxl,  Ymax7   ,  825,0,pxl
9380                 DATA 58,  Xmin8   ,   75,0,pxl,  Xmax8   ,  325,0,pxl,  Ymin8   ,  165,0,pxl,  Ymax8   ,  465,0,pxl
9390                 DATA 59,  Xmin9   ,  425,0,pxl,  Xmax9   ,  675,0,pxl,  Ymin9   ,  165,0,pxl,  Ymax9   ,  465,0,pxl
9400                 DATA 61,  Xdiv1   ,   10,0,"" ,  Ydiv1   ,    4,0,"" ,  Xdiv6   ,    4,0,"" ,  Ydiv6   ,    8,0,""
9410                 DATA 62,  Xdiv2   ,   10,0,"" ,  Ydiv2   ,    4,0,"" ,  Xdiv7   ,    5,0,"" ,  Ydiv7   ,    8,0,""
9420                 DATA 63,  Xdiv3   ,   10,0,"" ,  Ydiv3   ,    4,0,"" ,  Xdiv8   ,    4,0,"" ,  Ydiv8   ,    8,0,""
9430                 DATA 64,  Xdiv4   ,    4,0,"" ,  Ydiv4   ,    4,0,"" ,  Xdiv9   ,    4,0,"" ,  Ydiv9   ,    8,0,""
9440                 DATA 65,  Xdiv5   ,    4,0,"" ,  Ydiv5   ,    4,0,"" ,  ""      ,    0,0,"" ,  ""      ,    0,0,""
9450                 !     Y   *********X=1*********  *********X=2*********  *********X=3*********  *********X=4********
9460                 !         Delta   ,    0,4,°  ,  Beta    ,    0,4,°  ,  Cfreq   ,    0,0,Hz ,  Ofreq   ,    0,4,Hz
9470                 !         ""      ,    0,0,"" ,  Ujet/Ue ,    1,4,m/s,  ""      ,    0,0,"" ,  ""      ,    0,0,""
```

```
9480                   !     Y    *********X=1*********    *********X=2*********    *********X=3*********    *********X=4*********
9490                 SUBEND
9500 Array_print:    SUB Array_print(Array(*),Name$(*),Image$(*),Units$(*))
9510                   PRINT USING "#,5/"
9520                   FOR Y=1 TO SIZE(Array,1)
9530                     MAT SEARCH Array(Y,*),#LOC(<>0);L1
9540                     MAT SEARCH Name$(Y,*),#LOC(<>"");L2
9550                     IF L1+L2=0 AND L3=0 THEN 9790
9560                     L3=L1+L2
9570                     PRINT USING "#,28X"
9580                     FOR X=1 TO SIZE(Array,2)
9590                       SELECT Name$(Y,X)
9600                       CASE ""
9610                         PRINT USING "#,28X"
9620                       CASE "Date"
9630                         L$=DATE$(Array(Y,X))
9640                         L$=L$[1,2]&L$[4,6]&L$[8,11]
9650                         PRINT USING "#,10A,A,9A,X,3A,4X";TRIM$(Name$(Y,X)),"=",L$,Units$(Y,X)
9660                       CASE "Time"
9670                         L$=" "&TIME$(Array(Y,X))
9680                         PRINT USING "#,10A,A,9A,X,3A,4X";TRIM$(Name$(Y,X)),"=",L$,Units$(Y,X)
9690                       CASE ELSE
9700                         IF Image$(Y,X)="" THEN Image$(Y,X)="9D"
9710                         ON ERROR GOTO 9740
9720                         PRINT USING "#,10A,A,"&Image$(Y,X)&",X,3A,4X";TRIM$(Name$(Y,X)),"=",Array(Y,X),Units$(Y,X)
9730                         GOTO 9760
9740                         OFF ERROR
9750                         PRINT USING "#,10A,A,K,X,3A,4X";TRIM$(Name$(Y,X)),"=",Array(Y,X),Units$(Y,X)
9760                       END SELECT
9770                     NEXT X
9780                     PRINT
9790                   NEXT Y
9800                 SUBEND
9810 Change:         !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
9820 Change:         SUB Change(Type$,Array(*),Name$(*),Image$(*),Units$(*))
9830                   PRINTER IS CRT
9840                   FOR Y=1 TO SIZE(Array,1)
9850                     FOR Y1=Y TO SIZE(Array,1)
9860                       FOR X=1 TO SIZE(Array,2)
9870                         IF Name$(Y1,X)<>"" THEN 9920
9880                       NEXT X
9890                     NEXT Y1
9900                     CLEAR SCREEN
9910                     SUBEXIT
9920                     FOR Y2=Y1 TO SIZE(Array,1)
9930                       FOR X=1 TO SIZE(Array,2)
9940                         IF Name$(Y2,X)<>"" THEN 9970
9950                       NEXT X
9960                       GOTO 9980
9970                     NEXT Y2
9980                     FOR Y2=Y2 TO SIZE(Array,1)
9990                       FOR X=1 TO SIZE(Array,2)
10000                        IF Name$(Y2,X)<>"" THEN 10030
10010                      NEXT X
10020                    NEXT Y2
10030                    Y2=Y2-1
10040                    CLEAR SCREEN
10050                    CALL Display(Type$,Y1,Y2,Array(*),Name$(*),Image$(*),Units$(*))
10060                    Done=0
10070                    X=1
10080                    Y=Y1
10090                    ON KBD ALL,15 GOSUB Kbd
10100 Wait:              IF NOT Done THEN Wait
10110                    OFF KBD
10120                    CLEAR SCREEN
10130                    Y=Y2
10140                  NEXT Y
10150                  SUBEXIT
10160 Kbd:             CALL Update(Type$,X,Y,Y1,Y2,Done,Array(*),Name$(*),Image$(*),Units$(*))
10170                  RETURN
10180                SUBEND
10190 Display:       SUB Display(Type$,Y1,Y2,Array(*),Name$(*),Image$(*),Units$(*))
10200                  FOR Y=Y1 TO Y2
10210                    FOR X=1 TO SIZE(Array,2)
10220                      CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
10230                    NEXT X
10240                  NEXT Y
10250                  CALL Select(Type$,1,Y1,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
10260                SUBEND
10270 Select:        SUB Select(Type$,X,Y,Y1,Y2,C,Array(*),Name$(*),Image$(*),Units$(*))
```

```
10280                PRINT CHR$(128+C);TABXY(26*X-24,15+Y-Y1+1);
10290                PRINT RPT$(" ",23);TABXY(26*X-24,15+Y-Y1+1);
10300                IF Name$(Y,X)="" AND Array(Y,X)=0 THEN 10500
10310                Img$=Image$(Y,X)
10320                Unt$=Units$(Y,X)
10330                IF Image$(Y,X)="" THEN Img$="K"
10340                IF Units$(Y,X)="" THEN Unt$="   "
10350                SELECT Type$
10360                CASE "VALUES"
10370                    SELECT Name$(Y,X)
10380                    CASE "Date"
10390                    CASE "Time"
10400                    CASE ELSE
10410                        PRINT USING "#,10A,A,"&Img$&",X,3A";Name$(Y,X),":",Array(Y,X),Unt$
10420                    END SELECT
10430                CASE "NAMES"
10440                    PRINT USING "#,10A,A,8A";Name$(Y,X),":",Name$(Y,X)
10450                CASE "UNITS"
10460                    PRINT USING "#,10A,A,8A";Name$(Y,X),":",Units$(Y,X)
10470                CASE "IMAGES"
10480                    PRINT USING "#,10A,A,8A";Name$(Y,X),":",Image$(Y,X)
10490                END SELECT
10500                PRINT CHR$(128);
10510            SUBEND
10520 Update:    SUB Update(Type$,X,Y,Y1,Y2,Done,Array(*),Name$(*),Image$(*),Units$(*))
10530                DISABLE
10540                K$=KBD$
10550                IF K$="" THEN 11010
10560                SELECT NUM(K$[1,1])
10570                CASE 27                                                 ! ESC
10580                    Done=1
10590                CASE 255
10600                    CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
10610                    SELECT NUM(K$[2,2])
10620                    CASE 73,80                                          ! Break,Stop
10630                        PAUSE
10640                    CASE 124                                            ! Menu
10650                        Done=1
10660                    CASE 38                                             ! Select
10670                        CALL Select(Type$,X,Y,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
10680                        SELECT Type$
10690                        CASE "VALUES"
10700                            IF Name$(Y,X)="" THEN CALL Enter_string("Name for "&Name$(Y,X),Name$(Y,X),"K")
10710                            IF Image$(Y,X)="" THEN CALL Enter_string("Image for "&Name$(Y,X),Image$(Y,X),"K")
10720                            CALL Enter_value(Name$(Y,X),Array(Y,X),Image$(Y,X))
10730                        CASE "NAMES"
10740                            CALL Enter_string("Name for "&Name$(Y,X),Name$(Y,X),"K")
10750                        CASE "UNITS"
10760                            CALL Enter_string("Units for "&Name$(Y,X),Units$(Y,X),"K")
10770                        CASE "IMAGES"
10780                            CALL Enter_string("Image for "&Name$(Y,X),Image$(Y,X),"K")
10790                        END SELECT
10800                        CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
10810                        IF X=SIZE(Array,2) THEN Y=Y+1
10820                        X=X+1
10830                    CASE 60                                             ! Left
10840                        X=X-1
10850                    CASE 62                                             ! Right
10860                        X=X+1
10870                    CASE 94                                             ! Up
10880                        Y=Y-1
10890                    CASE 86                                             ! Down
10900                        Y=Y+1
10910                    CASE 92                                             ! First
10920                        X=1
10930                        Y=1
10940                    END SELECT
10950                    X=(X-1) MOD SIZE(Array,2)+1
10960                    Y=(Y-Y1+1-1) MOD (Y2-Y1+1)+Y1
10970                    IF X<1 THEN X=SIZE(Array,2)
10980                    IF Y<Y1 THEN Y=Y2
10990                    CALL Select(Type$,X,Y,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
11000                END SELECT
11010                ENABLE
11020                SUBEXIT
11030            SUBEND
11040 Table:     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
11050 Table:     SUB Table(Table(*))
11060                OPTION BASE 1
11070                REAL Mantisa(0:1023),Time(0:1023),Freq(0:1023)
```

```
11080          IF Table(32766) THEN SUBEXIT
11090          FOR Bin=0 TO 1023
11100              Mantisa(Bin)=Bin
11110          NEXT Bin
11120          Mantisa(0)=1
11130          Min=0
11140          FOR Fringes=0 TO 1
11150              FOR Exponent=0 TO 15
11160                  Max=Min+1023
11170                  IF Max=32767 THEN
11180                      Max=32766
11190                      REDIM Mantisa(0:1022),Time(0:1022),Freq(0:1022)
11200                  END IF
11210                  DISP Fringes,Exponent
11220                  MAT Time= Mantisa*(2^(Exponent-1)/500000000)
11230                  MAT Freq= (2^(4-Fringes))/Time
11240                  MAT Freq= Freq/(1000000)
11250                  MAT Table(Min:Max)= Freq
11260                  Min=Min+1024
11270              NEXT Exponent
11280          NEXT Fringes
11290      SUBEND
11300 Ctm:     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
11310 Ctm_ldv: SUB Ctm_ldv(Index(*),Theta1(*),Tun2ldv(*),Ldv2tun(*))
11320          OPTION BASE 1
11330          REAL Theta2(3,3)
11340          ! Correct Theta for angles in water
11350          MAT Theta2= Theta1
11360         !Theta2(2,1)=ASN(Index(1)/Index(3)*SIN(Theta2(2,1)))
11370         !Theta2(2,2)=ASN(Index(1)/Index(3)*SIN(Theta2(2,2)))+90
11380          ! Tun2Lvd converts tunnel coordinates to laser coordinates.
11390          Tun2ldv(1,1)=COS(Theta2(1,1))
11400          Tun2ldv(1,2)=COS(Theta2(1,2))
11410          Tun2ldv(1,3)=COS(Theta2(1,3))
11420          Tun2ldv(2,1)=COS(Theta2(2,1))
11430          Tun2ldv(2,2)=COS(Theta2(2,2))
11440          Tun2ldv(2,3)=COS(Theta2(2,3))
11450          Tun2ldv(3,1)=COS(Theta2(3,1))
11460          Tun2ldv(3,2)=COS(Theta2(3,2))
11470          Tun2ldv(3,3)=COS(Theta2(3,3))
11480          ! Ldv2tun converts laser coordinates to tunnel coordinates.
11490          MAT Ldv2tun= INV(Tun2ldv)
11500      SUBEND
11510 Ctm_mod: SUB Ctm_mod(Alpha1,Alpha2,Alpha3,Mod2tun(*),Tun2mod(*))
11520          OPTION BASE 1
11530          REAL T1(3,3),T2(3,3),T3(3,3),Abc(3),Abc1(3),Abc2(3),Temp(3,3)
11540          ! Define 1st coordinate transformation matrix for Mod2tun.
11550          T1(1,1)=COS(Alpha1)
11560          T1(1,2)=SIN(Alpha1)
11570          T1(1,3)=0
11580          T1(2,1)=-SIN(Alpha1)
11590          T1(2,2)=COS(Alpha1)
11600          T1(2,3)=0
11610          T1(3,1)=0
11620          T1(3,2)=0
11630          T1(3,3)=1
11640          ! Define 2nd coordinate transformation matrix for Mod2tun.
11650          T2(1,1)=1
11660          T2(1,2)=0
11670          T2(1,3)=0
11680          T2(2,1)=0
11690          T2(2,2)=COS(-Alpha2)
11700          T2(2,3)=SIN(-Alpha2)
11710          T2(3,1)=0
11720          T2(3,2)=-SIN(-Alpha2)
11730          T2(3,3)=COS(-Alpha2)
11740          ! Define 3rd coordinate transformation matrix for Mod2tun.
11750          Abc1(1)=1
11760          Abc1(2)=0
11770          Abc1(3)=0
11780          MAT Abc2= T1*Abc1
11790          MAT Abc= T2*Abc2
11800          T3(1,1)=Abc(1)*Abc(1)*(1-COS(-Alpha3))+COS(-Alpha3)
11810          T3(1,2)=Abc(2)*Abc(1)*(1-COS(-Alpha3))+Abc(3)*SIN(-Alpha3)
11820          T3(1,3)=Abc(3)*Abc(1)*(1-COS(-Alpha3))-Abc(2)*SIN(-Alpha3)
11830          T3(2,1)=Abc(1)*Abc(2)*(1-COS(-Alpha3))-Abc(3)*SIN(-Alpha3)
11840          T3(2,2)=Abc(2)*Abc(2)*(1-COS(-Alpha3))+COS(-Alpha3)
11850          T3(2,3)=Abc(3)*Abc(2)*(1-COS(-Alpha3))+Abc(1)*SIN(-Alpha3)
11860          T3(3,1)=Abc(1)*Abc(3)*(1-COS(-Alpha3))+Abc(2)*SIN(-Alpha3)
11870          T3(3,2)=Abc(2)*Abc(3)*(1-COS(-Alpha3))-Abc(1)*SIN(-Alpha3)
```

```
11880                T3(3,3)=Abc(3)*Abc(3)*(1-COS(-Alpha3))+COS(-Alpha3)
11890                ! Mod2tun converts model coordinates to tunnel coordinates.
11900                MAT Temp= T2*T1
11910                MAT Mod2tun= T3*Temp
11920                ! Tun2mod converts tunnel coordinates to model coordinates.
11930                MAT Tun2mod= INV(Mod2tun)
11940             SUBEND
11950 Ctm_tcs1:   SUB Ctm_tcs1(Tcs2tun(*),Tun2tcs(*))
11960                OPTION BASE 1
11970                REAL Nair,Nglass,Nwater
11980                REAL Flonaxis,Floffaxis,Bsonaxis,Bsoffaxis
11990                REAL Theta(4),Onaxis,Offaxis
12000                REAL Xon,Yon,Xoff,Yoff,X1,Y1,Y2
12010                REAL Ba,Bb,Xc,Yc
12020                REAL X(4),Yposition,Thickness
12030                INTEGER Offa,Offb,Ona,Onb,Beam,I,J
12040                Offa=1
12050                Offb=2
12060                Ona=3
12070                Onb=4
12080                Flonaxis=19.25
12090                Floffaxis=19.25
12100                Bsonaxis=60/25.4
12110                Bsoffaxis=60/25.4
12120                Thickness=1.25
12130                Onaxis=0.
12140                Offaxis=45.0
12150                Nair=1.00
12160                Nglass=1.43
12170                Nwater=1.33
12180                Yposition=0
12190                GOSUB Findstart
12200                Y1=Yon
12210                X1=Xoff
12220                Y2=Yoff
12230                Yposition=1
12240                GOSUB Findstart
12250                Y2=Yon-Y1+Y2
12260                MAT Tun2tcs= IDN
12270                Tun2tcs(2,2)=-(Yon-Y1)
12280                Tun2tcs(4,2)=-SQRT((Xoff-X1)^2+(Yoff-Y2)^2)
12290                Tun2tcs(4,4)=0
12300                MAT Tcs2tun= INV(Tun2tcs)
12310                Tcs2tun(4,2)=0
12320                MAT Tun2tcs= IDN
12330                MAT Tcs2tun= IDN
12340                SUBEXIT
12350 Findstart:     Theta(Offa)=Offaxis+ATN(Bsoffaxis/(2*Floffaxis))
12360                Theta(Offb)=Offaxis-ATN(Bsoffaxis/(2*Floffaxis))
12370                Theta(Ona)=Onaxis+ATN(Bsonaxis/(2*Flonaxis))
12380                Theta(Onb)=Onaxis-ATN(Bsonaxis/(2*Flonaxis))
12390                FOR Beam=Offa TO Onb
12400                    X(Beam)=-Yposition*TAN(ASN(Nair/Nwater*SIN(Theta(Beam))))-
                     Thickness*TAN(ASN(Nair/Nglass*SIN(Theta(Beam))))
12410                NEXT Beam
12420                Ba=-Thickness-X(Offa)/TAN(Theta(Offa))
12430                Bb=-Thickness-X(Offb)/TAN(Theta(Offb))
12440                Xc=(Bb-Ba)/(1/TAN(Theta(Offa))-1/TAN(Theta(Offb)))
12450                Yc=Xc/TAN(Theta(Offb))+Bb
12460                Xoff=Xc-Floffaxis*SIN(Offaxis)
12470                Yoff=Yc-Floffaxis*COS(Offaxis)
12480                Ba=-Thickness-X(Ona)/TAN(Theta(Ona))
12490                Bb=-Thickness-X(Onb)/TAN(Theta(Onb))
12500                Xc=(Bb-Ba)/(1/TAN(Theta(Ona))-1/TAN(Theta(Onb)))
12510                Yc=Xc/TAN(Theta(Onb))+Bb
12520                Xon=Xc-Flonaxis*SIN(Onaxis)
12530                Yon=Yc-Flonaxis*COS(Onaxis)
12540                RETURN
12550             SUBEND
12560 Ctm_tcs2:   SUB Ctm_tcs2(Tcs2tun(*),Tun2tcs(*))
12570                OPTION BASE 1
12580                REAL Nair,Nglass,Nwater
12590                REAL Floffaxis,Bsoffaxis
12600                REAL Theta(2),Offaxis
12610                REAL Xoff,Yoff,X1,Y1
12620                REAL Ba,Bb,Xc,Yc
12630                REAL X(2),Yposition,Thickness
12640                INTEGER Offa,Offb,Beam,I,J
12650                Offa=1
12660                Offb=2
```

```
12670                Floffaxis=19.5
12680                Bsoffaxis=60/25.4
12690                Thickness=1.25
12700                Offaxis=-13.2
12710                Nair=1.00
12720                Nglass=1.43
12730                Nwater=1.33
12740                Yposition=0
12750                GOSUB Findstart
12760                X1=Xoff
12770                Y1=Yoff
12780                Yposition=1
12790                GOSUB Findstart
12800                X2=Xoff
12810                Y2=Yoff
12820                Kx=(X2-X1)
12830                Ky=(Y2-Y1)
12840                MAT Tun2tcs= IDN
12850                Tun2tcs(1,2)=Kx
12860                Tun2tcs(2,2)=-Ky
12870                Tun2tcs(4,4)=0
12880                MAT Tcs2tun= INV(Tun2tcs)
12890                Tcs2tun(4,2)=0
12900                MAT Tun2tcs= IDN
12910                MAT Tcs2tun= IDN
12920                SUBEXIT
12930 Findstart:     Theta(Offa)=Offaxis+ATN(Bsoffaxis/(2*Floffaxis))
12940                Theta(Offb)=Offaxis-ATN(Bsoffaxis/(2*Floffaxis))
12950                FOR Beam=Offa TO Offb
12960                    X(Beam)=-Yposition*TAN(ASN(Nair/Nwater*SIN(Theta(Beam))))-
                     Thickness*TAN(ASN(Nair/Nglass*SIN(Theta(Beam))))
12970                NEXT Beam
12980                Ba=-Thickness-X(Offa)/TAN(Theta(Offa))
12990                Bb=-Thickness-X(Offb)/TAN(Theta(Offb))
13000                Xc=(Bb-Ba)/(1/TAN(Theta(Offa))-1/TAN(Theta(Offb)))
13010                Yc=Xc/TAN(Theta(Offb))+Bb
13020                Xoff=Xc-Floffaxis*SIN(Offaxis)
13030                Yoff=Yc-Floffaxis*COS(Offaxis)
13040                RETURN
13050            SUBEND
13060 Tcs8:      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
13070 Tcs8init:  SUB Tcs8init(@Tcs8)
13080                REAL I(1:8),C(1:8)
13090                ASSIGN @Tcs8 TO 9;BYTE,FORMAT OFF,EOL ""
13100                CONTROL 9,0;1
13110                CONTROL 9,3;9600
13120                CONTROL 9,4;31
13130                CONTROL 9,12;IVAL("EF",16)
13140                CONTROL 9,13;9600
13150                CONTROL 9,14;31
13160                OUTPUT @Tcs8 USING "K,/";"VI0"
13170                ENTER @Tcs8 USING "8(K)";I(*)
13180                IF SUM(I)<>8 THEN OUTPUT @Tcs8 USING "K,/";"SI0"
13190                OUTPUT @Tcs8 USING "K,/";"VC0"
13200                ENTER @Tcs8 USING "8(K)";C(*)
13210                IF SUM(C)<>8 THEN OUTPUT @Tcs8 USING "K,/";"SC0:1,"
13220               !OUTPUT @Tcs8 USING "K,/";"SC0:0,"
13230            SUBEND
13240 Tcs8set:   SUB Tcs8set(C$,@Tcs8)
13250                OPTION BASE 1
13260                DIM View(8,1),Set(8,2),Name$(8,1)[10],Image$(8,1)[10],Units$(8,1)[10]
13270                OUTPUT @Tcs8 USING "K,/";"V"&C$&"0"
13280                ENTER @Tcs8 USING "8(K)";View(*)
13290                READ Name$(*)
13300                MAT Image$= ("6D.3D")
13310                DATA X1,X2,Y1,Y2,Z1,Z2,A1,A2
13320                FOR Channel=1 TO 8
13330                    Set(Channel,1)=Channel
13340                    SELECT C$
13350                    CASE "P"
13360                        Name$(Channel,1)=Name$(Channel,1)&" (pos)"
13370                        Units$(Channel,1)="in"
13380                    CASE "U"
13390                        Name$(Channel,1)=Name$(Channel,1)&" (cpi)"
13400                        Units$(Channel,1)="cnt"
13410                    CASE "R"
13420                        Name$(Channel,1)=Name$(Channel,1)&" (cpr)"
13430                        Units$(Channel,1)="cnt"
13440                    CASE "V"
13450                        Name$(Channel,1)=Name$(Channel,1)&" (vel)"
```

```
13460                 Units$(Channel,1)="rev"
13470             CASE "A"
13480                 Name$(Channel,1)=Name$(Channel,1)&" (acc)"
13490                 Units$(Channel,1)="rev"
13500             CASE "+"
13510                 Name$(Channel,1)=Name$(Channel,1)&" (+LS)"
13520                 Units$(Channel,1)="    "
13530             CASE "-"
13540                 Name$(Channel,1)=Name$(Channel,1)&" (-LS)"
13550                 Units$(Channel,1)="    "
13560             CASE "S"
13570                 Name$(Channel,1)=Name$(Channel,1)&" (STALL)"
13580                 Units$(Channel,1)="    "
13590             CASE "H"
13600                 Name$(Channel,1)=Name$(Channel,1)&" (HS)"
13610                 Units$(Channel,1)="    "
13620             END SELECT
13630         NEXT Channel
13640         CALL Change("VALUES",View(*),Name$(*),Image$(*),Units$(*))
13650         SELECT C$
13660         CASE "P","U","R","V","A"
13670             MAT Set(*,2)= View(*,1)
13680             OUTPUT @Tcs8 USING 13690;"S"&C$,Set(*)
13690             IMAGE K,8(D,":",M6D.4D,","),/
13700         END SELECT
13710     SUBEND
13720 Tcs8read:     SUB Tcs8read(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Tcs2tun1(*),Tcs2tun2(*),Tun2mod(*))
13730         OUTPUT @Tcs8 USING "K,/";"VP0"
13740         ENTER @Tcs8 USING "8(K)";Tcs1(1),Tcs2(1),Tcs1(2),Tcs2(2),Tcs1(3),Tcs2(3),Tcs1(4),Tcs2(4)
13750         MAT Tun= Tcs2tun1*Tcs1
13760         REDIM Tun(1:3),Mod(1:3)
13770         MAT Mod= Tun2mod*Tun
13780         REDIM Tun(1:4),Mod(1:4)
13790         Mod(4)=0
13800         Tun(4)=0
13810         CALL Tcs8print(Mod(*),Tun(*),Tcs1(*),Tcs2(*))
13820     SUBEND
13830 Tcs8print:    SUB Tcs8print(Mod(*),Tun(*),Tcs1(*),Tcs2(*))
13840         PRINT CHR$(128);
13850         PRINT TABXY(50,1);"                                                 "
13860         PRINT TABXY(50,2);"          MOD        TUN        TCS1       TCS2   "
13870         PRINT TABXY(50,3);"                                                 "
13880         PRINT TABXY(50,4);
13890         PRINT USING "#,K,4(M3D.4D),X";" X:",Mod(1),Tun(1),Tcs1(1),Tcs2(1)
13900         PRINT TABXY(50,5);
13910         PRINT USING "#,K,4(M3D.4D),X";" Y:",Mod(2),Tun(2),Tcs1(2),Tcs2(2)
13920         PRINT TABXY(50,6);
13930         PRINT USING "#,K,4(M3D.4D),X";" Z:",Mod(3),Tun(3),Tcs1(3),Tcs2(3)
13940         PRINT TABXY(50,7);
13950         PRINT USING "#,K,4(M3D.4D),X";" A:",Mod(4),Tun(4),Tcs1(4),Tcs2(4)
13960         PRINT TABXY(50,8);"                                                 "
13970     SUBEND
13980 Tcs8move:     SUB
            Tcs8move(@Tcs8,Mod(*),Tun(*),Tcs1(*),Tcs2(*),Mod2tun(*),Tun2tcs1(*),Tun2tcs2(*),Side$,Coor$,Mode$,K,Mov
            ement)
13990         OPTION BASE 1
14000         DIM L$[100]
14010         REAL Move(8,2),I(8),C(8)
14020         IF Mode$="RELATIVE" THEN
14030             MAT Mod= (0)
14040             MAT Tun= (0)
14050             MAT Tcs1= (0)
14060             MAT Tcs2= (0)
14070         END IF
14080         SELECT Coor$
14090         CASE "MODEL"
14100             Mod(K)=Movement
14110             REDIM Tun(1:3),Mod(1:3)
14120             MAT Tun= Mod2tun*Mod
14130             REDIM Tun(1:4),Mod(1:4)
14140             IF POS(Side$,"Tx") THEN MAT Tcs1= Tun2tcs1*Tun
14150             IF POS(Side$,"Rx") THEN MAT Tcs2= Tun2tcs2*Tun
14160         CASE "TUNNEL"
14170             Tun(K)=Movement
14180             IF POS(Side$,"Tx") THEN MAT Tcs1= Tun2tcs1*Tun
14190             IF POS(Side$,"Rx") THEN MAT Tcs2= Tun2tcs2*Tun
14200         CASE "LASER"
14210             IF POS(Side$,"Tx") THEN Tcs1(K)=Movement
14220             IF POS(Side$,"Rx") THEN Tcs2(K)=Movement
14230         END SELECT
```

```
14240                FOR Channel=1 TO 8
14250                    Move(Channel,1)=Channel
14260                NEXT Channel
14270                Move(1,2)=Tcs1(1)
14280                Move(2,2)=Tcs2(1)
14290                Move(3,2)=Tcs1(2)
14300                Move(4,2)=Tcs2(2)
14310                Move(5,2)=Tcs1(3)
14320                Move(6,2)=Tcs2(3)
14330                Move(7,2)=Tcs1(4)
14340                Move(8,2)=Tcs2(4)
14350                SELECT Mode$
14360                CASE "ABSOLUTE"
14370                    OUTPUT @Tcs8 USING 14380;"MA",3,4,Move(3,2)
14380                    IMAGE K,1(D,D,":",K,",",),/
14390                    ENTER @Tcs8 USING "K";L$                ! Tcs1(2)
14400                    Tcs1(2)=VAL(L$)
14410                    ENTER @Tcs8 USING "K";L$                ! Tcs2(2)
14420                    Tcs2(2)=VAL(L$)
14430                CASE "RELATIVE"
14440                    OUTPUT @Tcs8 USING 14450;"MR",Move(*)
14450                    IMAGE K,8(D,":",S2D.5D,",",),/
14460                    ENTER @Tcs8 USING "8(K)";Tcs1(1),Tcs2(1),Tcs1(2),Tcs2(2),Tcs1(3),Tcs2(3),Tcs1(4),Tcs2(4)
14470                END SELECT
14480            SUBEND
14490 Tcs8view:  SUB Tcs8view(@Tcs8)
14500                OPTION BASE 1
14510                REAL View(8)
14520                C$="-+HS"
14530                CLEAR SCREEN
14540                PRINT TABXY(1,1);"        X1 X2 Y1 Y2 Z1 Z2 A1 A2"
14550                FOR I=1 TO 4
14560                    OUTPUT @Tcs8 USING "K,/";"V"&C$[I,I]&"0"
14570                    ENTER @Tcs8 USING "8(K)";View(*)
14580                    PRINT USING "AA,5X,8(3D)";"V"&C$[I,I],View(*)
14590                NEXT I
14600                BEEP
14610                GOTO 14540
14620            SUBEND
14630 Graph:     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
14640 Dump:      SUB Dump(G1,G2,Prt,Array(*),INTEGER Gs(*))
14650                OPTION BASE 1
14660                ALLOCATE INTEGER Ws(400,400)
14670                GSTORE Gs(*)
14680                KEY LABELS OFF
14690                OUTPUT Prt USING "#,@"
14700                FOR G=G1 TO G2
14710                    Xmin=Array(G+40,1)
14720                    Xmax=Array(G+40,2)
14730                    Ymin=Array(G+40,3)
14740                    Ymax=Array(G+40,4)
14750                    Xpix1=Array(G+50,1)-75
14760                    Xpix2=Array(G+50,2)+25
14770                    Ypix1=Array(G+50,3)-50
14780                    Ypix2=Array(G+50,4)+25
14790                    VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
14800                    WINDOW 0,1,0,1
14810                    CALL Bstore(Ws(*),(Xpix2-Xpix1)+1,(Ypix2-Ypix1)+1,3,0,1)
14820                    GCLEAR
14830                    CLEAR SCREEN
14840                    Xnew=100-Xpix1
14850                    Ynew=400-Ypix1
14860                    Xpix1=Xpix1+Xnew
14870                    Xpix2=Xpix2+Xnew
14880                    Ypix1=Ypix1+Ynew
14890                    Ypix2=Ypix2+Ynew
14900                    WINDOW 0,1,0,1
14910                    VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
14920                    CALL Bload(Ws(*),(Xpix2-Xpix1)+1,(Ypix2-Ypix1)+1,3,0,1)
14930                    CALL Gdump_colored(CRT,Prt,"NORMAL",180,"OFF","DITHER")
14940                    GLOAD Gs(*)
14950                NEXT G
14960                DEALLOCATE Ws(*)
14970            SUBEND
14980 Crt_init:  SUB Crt_init
14990                PLOTTER IS CRT,"INTERNAL"
15000                AREA PEN 0
15010                PEN 1
15020                PRINTER IS CRT
15030                PRINTALL IS CRT
```

```
15040                     KEY LABELS OFF
15050                 SUBEND
15060 Read_symbols:   SUB Read_symbols(Symbols(*))
15070                     OPTION BASE 1
15080                     REAL Symbol(20,3),Dot(2,3)
15090                     READ Dot(*)
15100                     FOR S=1 TO 5
15110                         READ Noc
15120                         REDIM Symbol(Noc,3)
15130                         READ Symbol(*)
15140                         MAT Symbols(S,1:Noc,*)= Symbol
15150                         MAT Symbols(S,Noc+1:Noc+2,*)= Dot
15160                         Symbols(S,0,1)=Noc+2
15170                     NEXT S
15180 Dot:                DATA     4.5, 7.5,-2,  4.5, 7.5,-1
15190 Square:             DATA 5,  0.5, 3.5,-2,  8.5, 3.5,-1,  8.5,11.5,-1,  0.5,11.5,-1,  0.5,3.5,-1
15200 Octagon:            DATA 9,  0.5, 5.5,-2,  2.5, 3.5,-1,  6.5, 3.5,-1,  8.5, 5.5,-1,  8.5,9.5,-1,  6.5,11.5,-1,  2.5,11.5,-
                          1,  0.5,9.5,-1,  0.5,5.5,-1
15210 Diamond:            DATA 5, -0.5, 7.5,-2,  4.5, 2.5,-1,  9.5, 7.5,-1,  4.5,12.5,-1, -0.5,7.5,-1
15220 Utriangle:          DATA 4,  0.5, 4.5,-2,  8.5, 4.5,-1,  4.5,13.5,-1,  0.5, 4.5,-1
15230 Dtriangle:          DATA 4,  0.5,10.5,-2,  8.5,10.5,-1,  4.5, 1.5,-1,  0.5,10.5,-1
15240                 SUBEND
15250 Setup_graph:    SUB Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
15260                     OPTION BASE 1
15270                     COM /Graph/
                          Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
15280                     MAT Wndw= Array(41:49,*)
15290                     MAT Vwprt= Array(51:59,*)
15300                     MAT Xdiv(1:5)= Array(61:65,1)
15310                     MAT Xdiv(6:9)= Array(61:64,3)
15320                     MAT Ydiv(1:5)= Array(61:65,2)
15330                     MAT Ydiv(6:9)= Array(61:64,4)
15340                     MAT Ximage$= Image$(41:49,1)
15350                     MAT Yimage$= Image$(41:49,3)
15360                     FOR G=1 TO 9
15370                         READ G,Xlabel$(G)
15380                         FOR I=1 TO SIZE(Legend$,2)
15390                             READ Legend$(G,I)
15400                         NEXT I
15410                         SELECT G
15420                         CASE 1 TO 5
15430                             Ylabel$(G)=""
15440                         CASE 6 TO 9
15450                             Ylabel$(G)=CHR$(NUM("X")+Paxis-1)
15460                         END SELECT
15470                         CALL Set_up(G,Symbols(*))
15480                     NEXT G
15490                     SUBEXIT
15500                     DATA 1, ""                                   ,   "",     "",     "","",""
15510                     DATA 2, ""                                   ,   "",     "",     "","",""
15520                     DATA 3, ""                                   ,   "",     "",     "","",""
15530                     DATA 4, ""                                   ,   "",     "",     "","",""
15540                     DATA 5, ""                                   ,   "",     "",     "","",""
15550                     DATA 6, "Velocities / Uinf"                  ,  "U",    "V",    "W","",""
15560                     DATA 7, "RMS / Uinf"                         , "U1",   "V1",   "W1","",""
15570                     DATA 8, "Shear Stress / Uinf^2"              ,"U1V1","V1W1", "W1U1","",""
15580                     DATA 9, "Tt:3R   Uinf:m/s   Uedge:m/s" ,  "Tt","Uinf","Uedge","",""
15590                 SUBEND
15600 Set_up:         SUB Set_up(G,Symbols(*))
15610                     OPTION BASE 1
15620                     COM /Graph/
                          Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
15630                     DIM L$[80]
15640                     ON ERROR CALL Error
15650                     PLOTTER IS CRT,"INTERNAL"
15660                     Black=-1
15670                     White=1
15680                     CSIZE 100*15/1023
15690                     Xmin=Wndw(G,1)
15700                     Xmax=Wndw(G,2)
15710                     Ymin=Wndw(G,3)
15720                     Ymax=Wndw(G,4)
15730                     Xpix1=Vwprt(G,1)
15740                     Xpix2=Vwprt(G,2)
15750                     Ypix1=Vwprt(G,3)
15760                     Ypix2=Vwprt(G,4)
15770                     Xstep=(Xmax-Xmin)/Xdiv(G)
15780                     Ystep=(Ymax-Ymin)/Ydiv(G)
15790                     Xpixel=(Xmax-Xmin)/(Xpix2-Xpix1)
15800                     Ypixel=(Ymax-Ymin)/(Ypix2-Ypix1)
```

```
15810                    AREA PEN Black
15820                    PEN White
15830                    GOSUB Back_ground
15840                    GOSUB Axes
15850                   !GOSUB Grid
15860                    GOSUB Plot_area
15870                    CLIP OFF
15880                    GOSUB Ylabel
15890                    GOSUB Xlabel
15900                    CALL Legend(G,Symbols(*))
15910                    OFF ERROR
15920                    SUBEXIT
15930 Back_ground:       VIEWPORT (Xpix1-75)/10.23,(Xpix2+25)/10.23,(Ypix1-33)/10.23,(Ypix2+6)/10.23
15940                    WINDOW -1.E+9,1.E+9,-1.E+9,1.E+9
15950                    MOVE 0,0
15960                    WINDOW 0,1,0,1
15970                    MOVE 0,0
15980                    RECTANGLE 1,1,FILL
15990                    RETURN
16000 Axes:              VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-6)/10.23,(Ypix1-1)/10.23
16010                    WINDOW Xmin,Xmax,1,0
16020                    AXES Xstep,2,Xmin,0,1,1,1
16030                    VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix2+1)/10.23,(Ypix2+6)/10.23
16040                    WINDOW Xmin,Xmax,0,1
16050                    AXES Xstep,2,Xmin,0,1,1,1
16060                    VIEWPORT (Xpix1-6)/10.23,(Xpix1-1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16070                    WINDOW 1,0,Ymin,Ymax
16080                    AXES 2,Ystep,0,Ymin,1,1,1
16090                    VIEWPORT (Xpix2+1)/10.23,(Xpix2+6)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16100                    WINDOW 0,1,Ymin,Ymax
16110                    AXES 2,Ystep,0,Ymin,1,1,1
16120                    RETURN
16130 Grid:              VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16140                    WINDOW Xmin,Xmax,Ymin,Ymax
16150                    LINE TYPE 4
16160                    GRID Xstep,Ystep,Xmin,Ymin
16170                    LINE TYPE 1
16180                    RETURN
16190 Plot_area:         VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
16200                    WINDOW Xmin,Xmax,Ymin,Ymax
16210                    RETURN
16220 Xlabel:            LORG 5
16230                    FOR X=Xmin TO Xmax+Xstep/100 STEP Xstep
16240                        MOVE X,Ymin-12*Ypixel
16250                        OUTPUT L$ USING Ximage$(G);X
16260                        LABEL TRIM$(L$)
16270                    NEXT X
16280                    MOVE (Xmin+Xmax)/2,Ymin-25*Ypixel
16290                    LABEL Xlabel$(G)
16300                    RETURN
16310 Ylabel:            LORG 8
16320                    Len=0
16330                    FOR Y=Ymin TO Ymax+Ystep/100 STEP Ystep
16340                        MOVE Xmin-5*Xpixel,Y
16350                        OUTPUT L$ USING Yimage$(G);Y
16360                        LABEL TRIM$(L$)
16370                        Len=MAX(Len,LEN(TRIM$(L$)))
16380                    NEXT Y
16390                    MOVE Xmin-(5+7*Len)*Xpixel,(Ymin+Ymax)/2
16400                    LABEL Ylabel$(G)
16410                    LORG 5
16420                    RETURN
16430                SUBEND
16440 Legend:        SUB Legend(G,Symbols(*))
16450                    OPTION BASE 1
16460                    COM /Graph/
                         Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
16470                    DIM Symbol(20,3)
16480                    VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
16490                    WINDOW Vwprt(G,1),Vwprt(G,2),Vwprt(G,3),Vwprt(G,4)
16500                    Black=-1
16510                    White=1
16520                    CSIZE 100*15/1023
16530                    AREA PEN -1       ! Black
16540                    PEN 1             ! White
16550                    LORG 2
16560                    Len=0
16570                    FOR S=1 TO SIZE(Legend$,2)
16580                        Len=MAX(LEN(Legend$(G,S)),Len)
16590                    NEXT S
```

```
16600                   FOR S=1 TO SIZE(Legend$,2)
16610                       IF LEN(Legend$(G,S))=0 THEN 16690
16620                       Noc=Symbols(S,0,1)
16630                       REDIM Symbol(Noc,3)
16640                       MAT Symbol= Symbols(S,1:Noc,*)
16650                       MOVE Vwprt(G,2)-7*Len-23,Vwprt(G,4)-15*S+5
16660                       SYMBOL Symbol(*),FILL,EDGE
16670                       MOVE Vwprt(G,2)-7*Len-10,Vwprt(G,4)-15*S+4
16680                       LABEL Legend$(G,S)
16690                   NEXT S
16700               SUBEND
16710 Lvdas:        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
16720 Lvdas_init:   SUB Lvdas_init(@Gpio)
16730                   ASSIGN @Gpio TO 12;WORD,FORMAT OFF,EOL ""
16740                   OUTPUT @Gpio USING "#,AA";"HP"
16750               SUBEND
16760 Lvdas_sample_a:SUB Lvdas_sample_a(@Lvdas,Channel,Symbol(*))
16770                   OPTION BASE 1
16780                   INTEGER Gx,Gy,Data(1000,4),G(128,102),Iv(1000)
16790                   DIM L$[80],V(1000),Vv(1000),T(1000),Wndw(4),Vwprt(4)
16800                   Black=-1
16810                   White=1
16820                   READ Wndw(*),Xdiv,Ydiv,Vwprt(*),Ximage$,Yimage$,Xlabel$,Ylabel$
16830                   !    Xmin,Xmax,Ymin,Ymax,Xdiv,Ydiv,Xpix1,Xpix2,Ypix1,Ypix2,Ximage$,Yimage$,Xlabel$,Ylabel$
16840                   DATA    0,.001,-5  ,   5,  10,  10,   75, 1235,  165,  825,  6D.4D,  6D.3D,t (sec),      V
16850                   CALL Set_up(Wndw(*),Vwprt(*),Xdiv,Ydiv,Xlabel$,Ylabel$,Ximage$,Yimage$)
16860                   GSTORE G(*)
16870                   PEN White
16880                   OUTPUT @Lvdas USING "#,AA";"DT"
16890                   OUTPUT @Lvdas USING "#,AA,W";"SC",Channel
16900                   OUTPUT @Lvdas USING "AA";"RM"
16910                   OUTPUT @Lvdas USING "W,W";IVAL("08F2",16),IVAL("0000",16)
16920                   OUTPUT @Lvdas USING "W,W";IVAL("08F2",16),IVAL("1F3F",16)
16930                   ENTER @Lvdas USING "#,W";Data(*)
16940                   OUTPUT @Lvdas USING "#,AA";"ET"
16950                   MAT T= Data(*,2)
16960                   MAT V= Data(*,4)
16970                   MAT V= V*(5./2.^15)
16980                   MAT Vv= V . V
16990                   Ave=SUM(V)/1000
17000                   Sdv=SQR(SUM(Vv)/1000-Ave*Ave)
17010                   MAT SEARCH V(*),MIN;Min
17020                   MAT SEARCH V(*),MAX;Max
17030                   Dif=Max-Min
17040                   GLOAD G(*)
17050                   MOVE Xmin+10*Xpixel,Ymax-20*Ypixel
17060                   LORG 2
17070                   LABEL USING "5(M5D.4D)";Ave,Sdv,Min,Max,Dif
17080                   Ave=Ave/5*2^15
17090                   Sdv=Sdv/5*2^15
17100                   Min=Min/5*2^15
17110                   Max=Max/5*2^15
17120                   Dif=Max-Min
17130                   LABEL
17140                   LABEL USING "2(M8D.1D),3(M10D)";Ave,Sdv,Min,Max,Dif
17150                   Time=0
17160                   LORG 5
17170                   CLIP ON
17180                   FOR I=1 TO 1000
17190                       PLOT Time,V(I)
17200                       SYMBOL Symbol(*),EDGE
17210                       MOVE Time,V(I)
17220                       PLOT Time,V(I)
17230                       Time=Time+T(I)*.0000001
17240                   NEXT I
17250                   GOTO 16880
17260                   SUBEXIT
17270               SUBEND
17280 Lvdas_average: SUB Lvdas_average(Table(*),INTEGER Data(*),REAL Vave,Vsdv,Tave,Tsdv)
17290                   OPTION BASE 1
17300                   REAL V(1000),Vv(1000),T(1000),Tt(1000)
17310                   N=SIZE(Data,1)
17320                   REDIM V(N),Vv(N),T(N),Tt(N)
17330                   Channel=Data(1,3)+1
17340                   SELECT Channel
17350                   CASE 1,2,3
17360                       FOR I=1 TO N
17370                           V(I)=Table(BINAND(32767,BINCMP(Data(I,4))))
17380                       NEXT I
17390                   CASE 4,5
```

```
17400                MAT V= Data(*,4)
17410                MAT V= V*(5/32768)
17420            CASE 6,7
17430                MAT V= (0)
17440            END SELECT
17450            MAT Vv= V . V
17460            MAT T= Data(*,2)
17470            MAT T= T/(10000000)
17480            MAT Tt= T . T
17490            Vave=SUM(V)/N
17500            Tave=SUM(T)/N
17510            Vsdv=SQR(ABS(SUM(Vv)/N-Vave*Vave))
17520            Tsdv=SQR(ABS(SUM(Tt)/N-Tave*Tave))
17530            MAT SEARCH Data(*,1),#LOC(<>0);Bad1
17540            MAT SEARCH Data(*,2),#LOC(<0);Bad2
17550           !PRINT USING 15300;Channel,Vave,Vsdv,Tave,Tsdv,Bad1,Bad2
17560            IMAGE 4D,2(MBD.4D),2(M2D.6D),10X,2(5D)
17570        SUBEND
17580 Lvdas_sample_c:SUB Lvdas_sample_c(@Lvdas,Channel,Table(*),REAL Vave,Vsdv,Tave,Tsdv)
17590            OPTION BASE 1
17600            INTEGER Data(1000,4)
17610            OUTPUT @Lvdas USING "#,AA";"DT"
17620            OUTPUT @Lvdas USING "#,AA,W";"SC",Channel
17630            OUTPUT @Lvdas USING "AA";"RM"
17640            OUTPUT @Lvdas USING "W,W";IVAL("08F0",16),IVAL("0000",16)
17650            OUTPUT @Lvdas USING "W,W";IVAL("08F0",16),IVAL("1F3F",16)
17660            ENTER @Lvdas USING "#,W";Data(*)
17670            OUTPUT @Lvdas USING "#,AA";"ET"
17680            CALL Lvdas_average(Table(*),Data(*),Vave,Vsdv,Tave,Tsdv)
17690        SUBEND
17700 Lvdas_take:    SUB Lvdas_take(@Lvdas,Atime,Ctime,INTEGER At_exp,Ct_exp,Cmask,Nsam)
17710            OPTION BASE 1
17720            COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
17730            INTEGER At1,At2,Ct1,Ct2
17740            DISP "Taking Data"
17750            CALL Convert2words(Atime*10000000,At1,At2)
17760            CALL Convert2words(Ctime*10000000,Ct1,Ct2)
17770            OUTPUT @Lvdas USING "AA,8(W)";"CS",At1,At2,Ct1,Ct2,At_exp,Ct_exp,Cmask,Nsam
17780            ENTER @Lvdas USING "#,W";Nsam
17790            IF Nsam=0 THEN SUBEXIT
17800            REDIM Raw(1:Nsam,1:10)
17810            ENTER @Lvdas USING "#,W";Raw(*)
17820        SUBEND
17830 Data_reduce:   SUB Data_reduce(INTEGER At_exp,Ct_exp,Nsam)
17840            OPTION BASE 1
17850            COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
17860            REDIM U(Nsam),V(Nsam),W(Nsam),A(Nsam),B(Nsam),I(Nsam),C(Nsam),Valid(Nsam)
17870            DISP "Reducing Data"
17880            MAT I= Raw(*,1)
17890            MAT C= Raw(*,2)
17900            MAT Valid= Raw(*,5)
17910            MAT U= Raw(*,6)
17920            MAT V= Raw(*,7)
17930            MAT W= Raw(*,8)
17940            MAT A= Raw(*,9)
17950            MAT B= Raw(*,10)
17960            FOR K=1 TO Nsam
17970                U(K)=Table(U(K))
17980                V(K)=Table(V(K))
17990 !              W(K)=Table(W(K))
18000            NEXT K
18010            MAT A= A*(5/32768)
18020            MAT B= B*(5/32768)
18030            MAT I= I*(1/2^At_exp/10)
18040            MAT C= C*(1/2^Ct_exp/10)
18050            MAT U= U . Valid
18060            MAT V= V . Valid
18070            MAT W= W . Valid
18080            MAT A= A . Valid
18090            MAT B= B . Valid
18100            MAT I= I . Valid
18110            MAT C= C . Valid !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
18120            MAT W= (0) !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
18130            MAT B= (0) !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
18140        SUBEND
18150 Data_xfer:     SUB Data_xfer(@Mac,Run,File,INTEGER N)
18160            OPTION BASE 1
18170            COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
18180            OUTPUT @Mac USING 18190;Run,File,N
18190            IMAGE K,"  ",K," ",K,/
```

```
18200                FOR K=1 TO N
18210                  OUTPUT @Mac USING 18220;K,Valid(K),DROUND(U(K),5),DROUND(V(K),5),DROUND(A(K),5)
18220                  IMAGE K,"   ",K," ",2(K,"      "),K,/
18230                NEXT K
18240                OUTPUT @Mac USING "@,/"
18250              SUBEND
18260 Data_clip:   SUB Data_clip(INTEGER Nsam,REAL Umin,Umax,Vmin,Vmax)
18270                OPTION BASE 1
18280                COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
18290                DISP "Clipping Histograms"
18300                FOR K=1 TO Nsam
18310                  MAT SEARCH U(*),LOC(<Umin);L,K
18320                  IF L<Nsam THEN Valid(L)=0
18330                  K=L
18340                NEXT K
18350                FOR K=1 TO Nsam
18360                  MAT SEARCH U(*),LOC(>Umax);L,K
18370                  IF L<Nsam THEN Valid(L)=0
18380                  K=L
18390                NEXT K
18400                FOR K=1 TO Nsam
18410                  MAT SEARCH V(*),LOC(<Vmin);L,K
18420                  IF L<Nsam THEN Valid(L)=0
18430                  K=L
18440                NEXT K
18450                FOR K=1 TO Nsam
18460                  MAT SEARCH V(*),LOC(>Vmax);L,K
18470                  IF L<Nsam THEN Valid(L)=0
18480                  K=L
18490                NEXT K
18500                MAT U= U . Valid
18510                MAT V= V . Valid
18520                MAT W= W . Valid
18530                MAT A= A . Valid
18540                MAT B= B . Valid
18550                MAT I= I . Valid
18560                MAT C= C . Valid
18570              SUBEND
18580 Data_aconvert: SUB Data_aconvert(Gain)
18590                DISP "Converting Data"
18600                OPTION BASE 1
18610                COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
18620                N=SIZE(Raw,1)
18630                DIM Mv(1000),Mvn(1000),Amvn(1000),Sum(1000)
18640                REDIM Mv(N),Mvn(N),Amvn(N),Sum(N)
18650                !     A0,        A1,        A2,       A3,        A4,         A5,           A6,            A7
18660                DATA 150,257.10163,-28.16138,6.064559,-.792687,.05708673,-.002103462,.00003110036
18670                MAT Mv= A*(1000/Gain)         !  Tt_mv=Tt_raw/Gain*1000
18680                MAT Sum= (0)
18690                MAT Mvn= (1)
18700                FOR K=0 TO 7
18710                  READ An
18720                  MAT Amvn= (An)*Mvn
18730                  MAT Sum= Sum+Amvn
18740                  MAT Mvn= Mvn . Mv
18750                NEXT K
18760                MAT A= Sum+(460)
18770              SUBEND
18780 Data_fconvert: SUB Data_fconvert(Array(*))
18790                OPTION BASE 1
18800                COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
18810                DIM Frng_spc(3),Brg_frq(3),Mix_frq(3),Mea_sgn(3),Brg_sgn(3),Mix_sgn(3)
18820                DISP "Converting Data"
18830                MAT Frng_spc= Array(25,1:3)
18840                MAT Brg_frq= Array(26,1:3)
18850                MAT Mix_frq= Array(27,1:3)
18860                MAT Mea_sgn= Array(28,1:3)
18870                MAT Brg_sgn= Array(29,1:3)
18880                MAT Mix_sgn= Array(30,1:3)
18890                MAT U= U*(Mea_sgn(1))
18900                MAT V= V*(Mea_sgn(2))
18910                MAT W= W*(Mea_sgn(3))
18920                MAT U= U+(Brg_sgn(1)*Brg_frq(1)+Mix_sgn(1)*Mix_frq(1))
18930                MAT V= V+(Brg_sgn(2)*Brg_frq(2)+Mix_sgn(2)*Mix_frq(2))
18940                MAT W= W+(Brg_sgn(3)*Brg_frq(3)+Mix_sgn(3)*Mix_frq(3))
18950                MAT U= U*(Frng_spc(1))
18960                MAT V= V*(Frng_spc(2))
18970                MAT W= W*(Frng_spc(3))
18980                MAT W= (0)
18990              SUBEND
```

```
19000 Data_sum:     SUB Data_sum(Sum(*),INTEGER N(*),Nsam)
19010                  OPTION BASE 1
19020                  COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
19030                  REAL Uu(1000),Vv(1000),Ww(1000),Aa(1000),Bb(1000),Ii(1000),Cc(1000)
19040                  REAL Uv(1000),Vw(1000),Wu(1000),Ab(1000),Ua(1000),Va(1000),Wa(1000)
19050                  REDIM Uu(Nsam),Vv(Nsam),Ww(Nsam),Aa(Nsam),Bb(Nsam),Ii(Nsam),Cc(Nsam)
19060                  REDIM Uv(Nsam),Vw(Nsam),Wu(Nsam),Ab(Nsam),Ua(1000),Va(1000),Wa(1000)
19070                  DISP "Summing Data"
19080                  !
19090                  MAT Uu= U . U
19100                  MAT Vv= V . V
19110                  MAT Ww= W . W
19120                  MAT Aa= A . A
19130                  MAT Bb= B . B
19140                  MAT Uv= U . V
19150                  MAT Vw= V . W
19160                  MAT Wu= W . U
19170                  MAT Ab= A . B
19180                  MAT Ua= U . A
19190                  MAT Va= V . A
19200                  MAT Wa= W . A
19210                  MAT Ii= I . I
19220                  MAT Cc= C . C
19230                  !
19240                  Sum(1,1)=SUM(U)
19250                  Sum(2,1)=SUM(V)
19260                  Sum(3,1)=SUM(W)
19270                  Sum(4,1)=SUM(A)
19280                  Sum(5,1)=SUM(B)
19290                  Sum(6,1)=SUM(I)
19300                  Sum(7,1)=SUM(C)
19310                  Sum(1,2)=SUM(Uu)
19320                  Sum(2,2)=SUM(Vv)
19330                  Sum(3,2)=SUM(Ww)
19340                  Sum(4,2)=SUM(Aa)
19350                  Sum(5,2)=SUM(Bb)
19360                  Sum(6,2)=SUM(Ii)
19370                  Sum(7,2)=SUM(Cc)
19380                  Sum(1,3)=SUM(Uv)
19390                  Sum(2,3)=SUM(Vw)
19400                  Sum(3,3)=SUM(Wu)
19410                  Sum(4,3)=SUM(Ab)
19420                  Sum(5,3)=SUM(Ua)
19430                  Sum(6,3)=SUM(Va)
19440                  Sum(7,3)=SUM(Wa)
19450                  MAT N= (SUM(Valid))
19460                  N(3,1)=0
19470                  N(5,1)=0
19480                  N(3,2)=0
19490                  N(5,2)=0
19500                  N(2,3)=0
19510                  N(3,3)=0
19520                  N(4,3)=0
19530                  N(6,3)=0
19540                  N(7,3)=0
19550               SUBEND
19560 Data_calc:    SUB Data_calc(INTEGER N(*),REAL
                       Sum(*),U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
19570                  DISP "Calculating Results"
19580                  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
19590                  U=0
19600                  V=0
19610                  W=0
19620                  A=0
19630                  B=0
19640                  I=0
19650                  C=0
19660                  IF N(1,1) THEN U=Sum(1,1)/N(1,1)
19670                  IF N(2,1) THEN V=Sum(2,1)/N(2,1)
19680                  IF N(3,1) THEN W=Sum(3,1)/N(3,1)
19690                  IF N(4,1) THEN A=Sum(4,1)/N(4,1)
19700                  IF N(5,1) THEN B=Sum(5,1)/N(5,1)
19710                  IF N(6,1) THEN I=Sum(6,1)/N(6,1)
19720                  IF N(7,1) THEN C=Sum(7,1)/N(7,1)
19730                  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
19740                  U1=0
19750                  V1=0
19760                  W1=0
19770                  A1=0
19780                  B1=0
```

```
19790              I1=0
19800              C1=0
19810              IF N(1,2) THEN U1=SQR(ABS(Sum(1,2)/N(1,2)-U*U))
19820              IF N(2,2) THEN V1=SQR(ABS(Sum(2,2)/N(2,2)-V*V))
19830              IF N(3,2) THEN W1=SQR(ABS(Sum(3,2)/N(3,2)-W*W))
19840              IF N(4,2) THEN A1=SQR(ABS(Sum(4,2)/N(4,2)-A*A))
19850              IF N(5,2) THEN B1=SQR(ABS(Sum(5,2)/N(5,2)-B*B))
19860              IF N(6,2) THEN I1=SQR(ABS(Sum(6,2)/N(6,2)-I*I))
19870              IF N(7,2) THEN C1=SQR(ABS(Sum(7,2)/N(7,2)-C*C))
19880              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
19890              U1v1=0
19900              V1w1=0
19910              W1u1=0
19920              A1b1=0
19930              U1a1=0
19940              V1a1=0
19950              W1a1=0
19960              IF N(1,3) THEN U1v1=Sum(1,3)/N(1,3)-U*V
19970              IF N(2,3) THEN V1w1=Sum(2,3)/N(2,3)-V*W
19980              IF N(3,3) THEN W1u1=Sum(3,3)/N(3,3)-W*U
19990              IF N(4,3) THEN A1b1=Sum(4,3)/N(4,3)-A*B
20000              IF N(5,3) THEN U1a1=Sum(5,3)/N(5,3)-U*A
20010              IF N(6,3) THEN V1a1=Sum(6,3)/N(6,3)-V*A
20020              IF N(7,3) THEN W1a1=Sum(7,3)/N(7,3)-W*A
20030              !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
20040          SUBEND
20050 Data_trnsfrm: SUB Data_trnsfrm(REAL K(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1)
20060              OPTION BASE 1
20070              REAL Vabc(3),Vuvw(3),Kabc(3,3),First
20080              REAL Ku(3,3),Kv(3,3),Kw(3,3),Ktu(3,3),Ktv(3,3),Ktw(3,3)
20090              REAL Kuu(3,3),Kvv(3,3),Kww(3,3),Kuv(3,3),Kvw(3,3),Kwu(3,3)
20100              REAL Ku1u1(3,3),Kv1v1(3,3),Kw1w1(3,3),Ku1v1(3,3),Kv1w1(3,3),Kw1u1(3,3)
20110              DISP "Transforming Results"
20120              Vabc(1)=U
20130              Vabc(2)=V
20140              Vabc(3)=W
20150              Kabc(1,1)=U1*U1
20160              Kabc(1,2)=U1v1
20170              Kabc(1,3)=W1u1
20180              Kabc(2,1)=U1v1
20190              Kabc(2,2)=V1*V1
20200              Kabc(2,3)=V1w1
20210              Kabc(3,1)=W1u1
20220              Kabc(3,2)=V1w1
20230              Kabc(3,3)=W1*W1
20240              MAT Vuvw= K*Vabc
20250             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"K     ",K(*)
20260              U=Vuvw(1)
20270              V=Vuvw(2)
20280              W=Vuvw(3)
20290              FOR I=1 TO 3
20300                  FOR J=1 TO 3
20310                      Ku(I,J)=K(1,I)
20320                      Kv(I,J)=K(2,I)
20330                      Kw(I,J)=K(3,I)
20340                  NEXT J
20350              NEXT I
20360             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Ku    ",Ku(*)
20370             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kv    ",Kv(*)
20380             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kw    ",Kw(*)
20390              MAT Ktu= TRN(Ku)
20400              MAT Ktv= TRN(Kv)
20410              MAT Ktw= TRN(Kw)
20420             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Ktu   ",Ktu(*)
20430             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Ktv   ",Ktv(*)
20440             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Ktw   ",Ktw(*)
20450              MAT Kuu= Ku . Ktu
20460              MAT Kvv= Kv . Ktv
20470              MAT Kww= Kw . Ktw
20480              MAT Kuv= Ku . Ktv
20490              MAT Kvw= Kv . Ktw
20500              MAT Kwu= Kw . Ktu
20510             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kuu   ",Kuu(*)
20520             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kvv   ",Kvv(*)
20530             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kww   ",Kww(*)
20540             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kuv   ",Kuv(*)
20550             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kvw   ",Kvw(*)
20560             !OUTPUT PRT USING "6A,/,3(3(5DZ.5D),/),/";"Kwu   ",Kwu(*)
20570              MAT Ku1u1= Kuu . Kabc
20580              MAT Kv1v1= Kvv . Kabc
```

```
20590                 MAT Kw1w1= Kww . Kabc
20600                 MAT Ku1v1= Kuv . Kabc
20610                 MAT Kv1w1= Kvw . Kabc
20620                 MAT Kw1u1= Kwu . Kabc
20630                 U1u1=SUM(Ku1u1)
20640                 V1v1=SUM(Kv1v1)
20650                 W1w1=SUM(Kw1w1)
20660                 U1v1=SUM(Ku1v1)
20670                 V1w1=SUM(Kv1w1)
20680                 W1u1=SUM(Kw1u1)
20690                 U1=SQR(ABS(U1u1))
20700                 V1=SQR(ABS(V1v1))
20710                 W1=SQR(ABS(W1w1))
20720               SUBEND
20730 Data_print:   SUB Data_print(Axis,Pos,INTEGER Nsam,C$,REAL
                      U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1)
20740                 IF C$="LDV" OR C$="TUN" THEN SUBEXIT
20750                 DISP "Printing Results"
20760                 ON ERROR CALL Error
20770                 PRINTER IS PRT;WIDTH 144
20780                 PRINT CHR$(27)&"&k2S"&CHR$(27)&"&l9D";
20790                 L$=CHR$(NUM("X")+Axis-1)
20800                 SELECT C$
20810                 CASE "MHz","MOD"
20820                      PRINT USING 20920;L$,Pos,U,U1,U1v1
20830                      PRINT USING 20950;A,A1,A1b1,U1a1
20840                      PRINT USING 20930;"N",Nsam,V,V1,V1w1
20850                      PRINT USING 20960;B,B1,I1,V1a1
20860                      PRINT USING 20940;C$[1,3],W,W1,W1u1
20870                      PRINT USING 20970;C,I,C1,W1a1
20880                      IF C$<>"MOD" THEN PRINT
20890                 END SELECT
20900                 PRINTER IS CRT
20910                 OFF ERROR
20920                 IMAGE #,8X, A,"=",3D.4D,"    U=",5D.3D,"    U1=",5D.3D,"    U1V1=",8D.2D
20930                 IMAGE #,8X, A,"=",     8D,"    V=",5D.3D,"    V1=",5D.3D,"    V1W1=",8D.2D
20940                 IMAGE #,8X,3A,         7X,"    W=",5D.3D,"    W1=",5D.3D,"    W1U1=",8D.2D
20950                 IMAGE    "   A =",5D.3D,"   A1 =",5D.3D,"   A1B1=",6D.2D,"   U1A1=",7D.2D
20960                 IMAGE    "   B =",5D.3D,"   B1 =",5D.3D,"   IAT1=",6D.2D,"   V1A1=",7D.2D
20970                 IMAGE    "   CT=",5D.3D,"   IAT=",5D.3D,"   CT1 =",6D.2D,"   W1A1=",7D.2D
20980               SUBEND
20990 Data_plot:    SUB Data_plot(Array(*),Symbols(*),G,Y,P1,P2,P3,Scale,INTEGER N1,N2,N3)
21000                 OPTION BASE 1
21010                 DIM Wndw(4),Vwprt(4),Symbol(20,3)
21020                 DISP "Ploting Results"
21030                 AREA PEN -1
21040                 PEN 1
21050                 MAT Wndw= Array(40+G,*)
21060                 MAT Vwprt= Array(50+G,*)
21070                 VIEWPORT Vwprt(1)/10.23,Vwprt(2)/10.23,Vwprt(3)/10.23,Vwprt(4)/10.23
21080                 WINDOW Wndw(1),Wndw(2),Wndw(3),Wndw(4)
21090                 CLIP ON
21100                 FOR I=0 TO 2
21110                      IF I=0 AND N1=0 THEN 21300
21120                      IF I=1 AND N2=0 THEN 21300
21130                      IF I=2 AND N3=0 THEN 21300
21140                      Sy=I+1
21150                      Noc=Symbols(Sy,0,1)
21160                      REDIM Symbol(Noc,3)
21170                      MAT Symbol= Symbols(Sy,1:Noc,*)
21180                      SELECT I
21190                      CASE 0
21200                           X=P1*Scale
21210                      CASE 1
21220                           X=P2*Scale
21230                      CASE 2
21240                           X=P3*Scale
21250                      END SELECT
21260                      Xm=MIN(MAX(X,Wndw(1)),Wndw(2))
21270                      Ym=MIN(MAX(Y,Wndw(3)),Wndw(4))
21280                      MOVE Xm,Ym
21290                      SYMBOL Symbol(*),FILL,EDGE
21300                 NEXT I
21310               SUBEND
21320 Histo:        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
21330 Rt_histo:     SUB Rt_histo(@Lvdas,Symbols(*),Repeat)
21340                 OPTION BASE 1
21350                 COM /Graph/
                      Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
21360                 INTEGER Histo(1000,3),Nplots,Nbins,F1,F2,A1,A2
```

```
21370              REAL Nnew,Nold,N(5)
21380              OUTPUT @Lvdas USING "AA";"CA"
21390              FOR Channel=1 TO 5
21400                CALL Set_up(Channel,Symbols(*))
21410              NEXT Channel                                        ! Atime=.1 seconds
21420              CALL Convert2words(.1*10000000,A1,A2)
21430              ON KBD GOSUB Hdone
21440              REPEAT
21450                FOR Channel=1 TO 5
21460                  G=Channel
21470                  SELECT Channel
21480                  CASE 1,2
21490                    Min=0
21500                    Bin=20
21510                    Ww=2^Bin
21520                    Kw=1000000
21530                    CALL Convert2words(Min,F1,F2)
21540                  CASE 4
21550                    Min=-5
21560                    Bin=10
21570                    Ww=2^Bin
21580                    F1=-1
21590                    F2=-32768
21600                    Kw=32768/5
21610                  CASE ELSE
21620                    GOTO 21880
21630                  END SELECT
21640 Hsend:             OUTPUT @Lvdas USING "AA,6(W)";"TH",F1,F2,Bin,A1,A2,Channel
21650 Henter:            ENTER @Lvdas USING "#,W";Nbins
21660                  IF Nbins>0 THEN
21670                    REDIM Histo(Nbins,3)
21680                    ENTER @Lvdas USING "#,W";Histo(*)
21690                  END IF
21700                  ENTER @Lvdas USING "#,W";Nnew,Nold
21710 Hplot:             VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
21720                  WINDOW Kw*Wndw(G,1),Kw*Wndw(G,2),Wndw(G,3),Wndw(G,4)
21730                  Xpixel=Kw*(Wndw(Channel,2)-Wndw(Channel,1))/(Vwprt(Channel,2)-Vwprt(Channel,1))
21740                  N1=N(Channel)
21750                  N2=N(Channel)-Nold+Nnew
21760                  N(Channel)=N(Channel)-Nold+Nnew
21770                  FOR I=1 TO Nbins
21780                    Old=MIN(Histo(I,3),Wndw(Channel,4))
21790                    New=MIN(Histo(I,2),Wndw(Channel,4))
21800                    AREA PEN SGN(New-Old)
21810                    X1=Histo(I,1)*Ww+Min*Kw
21820                    X2=Ww
21830                    Y1=Old
21840                    Y2=New-Old
21850                    MOVE X1,Y1
21860                    RECTANGLE X2-Xpixel,Y2,FILL
21870                  NEXT I
21880                NEXT Channel
21890              UNTIL KBD$<>"" OR NOT Repeat
21900              SUBEXIT
21910 Hdone:       Done=1
21920              RETURN
21930            SUBEND !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
21940 Histo:     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
21950 Pt_histo:  SUB Pt_histo(Symbols(*),Run,File,Pos,INTEGER Nsam)
21960              OPTION BASE 1
21970              COM /Data/ INTEGER Raw(*),Valid(*),REAL Table(*),U(*),V(*),W(*),A(*),B(*),I(*),C(*)
21980              COM /Graph/
           Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
21990              INTEGER Histo(0:100)
22000              REAL Data(1000)
22010              REDIM Data(Nsam)
22020              FOR Channel=5 TO 1 STEP -1
22030                G=Channel
22040                IF Channel=1 THEN MAT Data= U
22050                IF Channel=2 THEN MAT Data= V
22060                IF Channel=4 THEN MAT Data= A
22070                SELECT Channel
22080                CASE 1,2,4
22090                  CALL Set_up(Channel,Symbols(*))
22100 Hsort:             Xmin=Wndw(Channel,1)
22110                  Xmax=Wndw(Channel,2)
22120                  Xwin=(Xmax-Xmin)/100
22130                  MAT Data= Data-(Xmin)
22140                  MAT Data= Data/((Xmax-Xmin)/100)
22150                  MAT Histo= (0)
```

```
22160                         FOR K=1 TO Nsam
22170                             L=MAX(MIN(Data(K),100),0)
22180                             Histo(L)=Histo(L)+1
22190                         NEXT K
22200 Hplot:                  VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
22210                         WINDOW 0,100,Wndw(G,3),Wndw(G,4)
22220                         Xpixel=(100-0)/(Vwprt(Channel,2)-Vwprt(Channel,1))
22230                         MOVE 55,70
22240                         IF G=2 THEN LABEL USING "2A,2D.2D";"R=",Run
22250                         IF G=1 THEN LABEL USING "2A,2D.3D";"Y=",Pos
22260                         IF G=4 THEN LABEL USING "2A,2D   ";"F=",File
22270                         FOR K=0 TO 100
22280                             IF Histo(K) THEN
22290                                 MOVE K-.5,0
22300                                 AREA PEN SGN(1)
22310                                 RECTANGLE 1-Xpixel,Histo(K),FILL
22320                             END IF
22330                         NEXT K
22340                     END SELECT
22350                 NEXT Channel
22360                 SUBEXIT
22370             SUBEND
22380 Misc:       !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
22390 Convert2words: SUB Convert2words(Real,INTEGER High,Low)
22400                 Hex$=DVAL$(Real,16)
22410                 High=IVAL(Hex$[1,4],16)
22420                 Low=IVAL(Hex$[5,8],16)
22430             SUBEND
22440 Temp:       SUB Temp(Mach,Mv,Ts,Tt)
22450                 !     A0,       A1,        A2,       A3,       A4,        A5,          A6,           A7
22460                 DATA 150,257.10163,-28.16138,6.064559,-.792687,.05708673,-.002103462,.00003110036
22470                 Tt=0
22480                 FOR I=0 TO 7
22490                     READ K
22500                     Tt=Tt+K*Mv^I
22510                 NEXT I
22520                 Tt=Tt+460
22530                 Ts=.09259*Tt
22540                 IF Mach<>7 THEN BEEP
22550                 IF Mach<>7 THEN PAUSE
22560             SUBEND
22570 Error:      SUB Error
22580                 BEEP
22590                 DISP ERRM$
22600                 OUTPUT PRT;ERRM$
22610                 Prt=VAL(SYSTEM$("PRINTER IS"))
22620                 PRINTER IS CRT
22630                 PRINT TABXY(95,1);ERRM$
22640                 PRINTER IS Prt
22650                 ERROR SUBEXIT
22660             SUBEND
22670 Fix:        SUB Fix(Array(*),Name$(*),Image$(*),Units$(*))
22680                 OPTION BASE 1
22690                 Run=Array(3,1)
22700                 SELECT INT(Run)
22710                 CASE ELSE
22720                     Image$(3,1)="6D.2D"          ! Run
22730                     Array(3,1)=INT(Run)+.01      ! Run
22740                     Array(21,1)=.3125            ! UBeamSpc
22750                     Array(21,2)=.34375           ! VBeamSpc
22760                     Array(21,3)=.3125            ! WBeamSpc
22770                     Array(22,1)=30               ! UFoclLen
22780                     Array(22,2)=30               ! VFoclLen
22790                     Array(22,3)=30               ! WFoclLen
22800                     Array(23,1)=2*ATN(Array(21,1)/2/Array(22,1))   ! UFrngSpc
22810                     Array(23,2)=2*ATN(Array(21,2)/2/Array(22,2))   ! VFrngSpc
22820                     Array(23,3)=2*ATN(Array(21,3)/2/Array(22,3))   ! WFrngSpc
22830                     Array(28,1)=-1               ! UMeaSgn
22840                     Array(29,1)=1                ! UBrgSgn
22850                     Array(30,1)=1                ! UMixSgn
22860                     Array(41,1)=0                ! Xmin1
22870                     Array(41,2)=100              ! Xmax1
22880                     Array(42,1)=0                ! Xmin2
22890                     Array(42,2)=100              ! Xmax2
22900                     Array(43,1)=0                ! Xmin3
22910                     Array(43,2)=100              ! Xmax3
22920                     Array(44,1)=-5               ! Xmin4
22930                     Array(44,2)=5                ! Xmax4
22940                     Array(45,1)=-5               ! Xmin5
22950                     Array(45,2)=5                ! Xmax5
```

```
22960                    Array(61,1)=5                 ! Xdiv1
22970                    Array(62,1)=5                 ! Xdiv2
22980                    Array(63,1)=5                 ! Xdiv3
22990                    Array(64,1)=4                 ! Xdiv4
23000                    Array(65,1)=4                 ! Xdiv5
23010                    !
23020                    Array(46,4)=4                 ! Ymax6
23030                    Array(47,4)=4                 ! Ymax7
23040                    Array(48,4)=4                 ! Ymax8
23050                    Array(49,4)=4                 ! Ymax9
23060                    Array(49,1)=0                 ! Xmin9
23070                    Array(49,2)=2000              ! Xmax9
23080                    Array(61,4)=8                 ! Ydiv6
23090                    Array(62,4)=8                 ! Ydiv7
23100                    Array(63,4)=8                 ! Ydiv8
23110                    Array(64,4)=8                 ! Ydiv9
23120                    Array(64,3)=4                 ! Xdiv9
23130                    !
23140                    Array(35,1)=8                 ! UFreqMin
23150                    Array(36,1)=40                ! UFreqMax
23160                    Array(35,2)=20                ! VFreqMin
23170                    Array(36,2)=55                ! VFreqMax
23180                    Array(35,3)=10                ! WFreqMin
23190                    Array(36,3)=70                ! WFreqMax
23200                    Array(36,4)=1                 ! Clip
23210                    !
23220                    Name$(35,1)="UFreqMin"        ! UFreqMin
23230                    Name$(36,1)="UFreqMax"        ! UFreqMax
23240                    Name$(35,2)="VFreqMin"        ! VFreqMin
23250                    Name$(36,2)="VFreqMax"        ! VFreqMax
23260                    Name$(35,3)="WFreqMin"        ! WFreqMin
23270                    Name$(36,3)="WFreqMax"        ! WFreqMax
23280                    Name$(36,4)="Clip"            ! Clip
23290                    !
23300                    Units$(35,1)="MHz"            ! UFreqMin
23310                    Units$(36,1)="MHz"            ! UFreqMax
23320                    Units$(35,2)="MHz"            ! VFreqMin
23330                    Units$(36,2)="MHz"            ! VFreqMax
23340                    Units$(35,3)="MHz"            ! WFreqMin
23350                    Units$(36,3)="MHz"            ! WFreqMax
23360                    Units$(36,4)=""               ! Clip
23370                    !
23380                    Image$(35,1)="4D.4D"          ! UFreqMin
23390                    Image$(36,1)="4D.4D"          ! UFreqMax
23400                    Image$(35,2)="4D.4D"          ! VFreqMin
23410                    Image$(36,2)="4D.4D"          ! VFreqMax
23420                    Image$(35,3)="4D.4D"          ! WFreqMin
23430                    Image$(36,3)="4D.4D"          ! WFreqMax
23440                    Image$(36,4)="9D"             ! Clip
23450                    !
23460                END SELECT
23470            SUBEND
23480 Scale:     SUB Scale(G)
23490                OPTION BASE 1
23500                COM /Graph/
                 Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
23510                VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
23520                WINDOW Wndw(G,1),Wndw(G,2),Wndw(G,3),Wndw(G,4)
23530            SUBEND
23540 Purge:     SUB Purge(System$,Data$)
23550                OPTION BASE 1
23560                DIM F$(400)[80]
23570                MASS STORAGE IS Data$
23580                CAT TO F$(*);NAMES
23590                MAT SEARCH F$(*),LOC(="");N
23600                N=N-1
23610                IF N>0 THEN
23620                    REDIM F$(N)
23630                    FOR I=1 TO N
23640                        IF F$(I)[1,4]="R.01" THEN
23650                            PURGE F$(I)&"<TKM>"
23660                            DISP F$(I)&"<TKM>"
23670                        END IF
23680                    NEXT I
23690                END IF
23700                MASS STORAGE IS System$
23710            SUBEND
```

APPENDIX B

REVISED SOFTWARE CODE LISTING.

APPENDIX B

Revised Software Code Listing.

TABLE OF CONTENTS

Hard Disk Directory Catalog Listing.

:CS80, 1400, 0, 0

VOLUME LABEL: B9826

FILE NAME	PRO	TYPE	REC/FILE	BYTE/REC	ADDRESS	DATE	TIME
SYSB60		SYSTM	3388	256	32	17-Jul-91	13:10
CDUMP6		PROG	44	256	3420	17-Jul-91	13:10
BPLOT6		PROG	40	256	3464	17-Jul-91	13:10
AUTOST		PROG	10	256	3504	17-Jul-91	13:10
ARRAY		BDAT	50	256	3515	17-Jul-91	13:11
KEYS		BDAT	4	256	3566	17-Jul-91	13:11
COPY		PROG	25	256	3570	17-Jul-91	13:11
3.5'HWT92		PROG	816	256	3967	29-Mar-92	13:45

```
                                                                                                  3.5'HWT92
100 Main: !---------------------------------------------------------------------------------------!
105       !                                                    Property of COMPLERE INC.          !
110       !              NASA AMES RESEARCH CENTER                 Proprietary software           !
115       !          3.5 FOOT HYPERSONIC WIND TUNNEL            Copyright March 29, 1992          !
120       !                                                Developed by:  T. Kevin McDevitt       !
125       !            Laser Doppler Velocimeter Test                                              !
130       !                                                                                        !
135       !---------------------------------------------------------------------------------------!
140       !
145       ! PROGRAM DESCRIPTION:
150       !
155       !      This program provides the capability to acquire simultaneous Laser Doppler Velocimeter (LDV), Stagnation
160       !   Temperature, and analog voltage data at user selectable traverse controlled probe volume positions within
165       !   the hypersonic wind tunnel flow.
170       !      The Laser Velocimeter Data Acquisition System (LVDAS) is used to sample the LDV and analog voltage
175       !   data simultaneously with a coincidence criterion being applied to LDV incoming data.  The LVDAS also generates
180       !   inter-arrival times and coincidence times.
185       !      The measured LDV data provide the necessary frequency information from which two components of flow
190       !   velocities can be determined.  These velocities are measured directly in "TUNNEL" coordinates.  A coordinate
195       !   system transformation is applied to these measured velocities to obtain velocities in and "MODEL" coordinates
200       !   if the model is at angles of attack, yaw, and/or roll.
205       !      The Traverse Control System (TCS8) is used to precisely move the LDV probe volume within the tunnel and
210       !   about the model.  The TCS8 provides three axes, plus one auxiliary axis, of traverse capability for both the
215       !   transmitting (Tx) and receiving (Rx) side optical packages.  The Tx and Rx side traverses can be moved
220       !   independently to achieve laser alignment or they can be moved together to maintain laser alignment.
225       !      The TCS8 will give the traverse positions in "TUNNEL" coordinates where one inch of commanded movement will
230       !   yield one inch of movement on the traverse slides.  This will also yield one inch of movement of the probe
235       !   volume crossover point within the tunnel.  However, the traverse positions in "TUNNEL" coordinates will differ
240       !   from positions in "MODEL" coordinates if the model is at angles of attack, yaw, and/or roll.  Therefore, a
245       !   coordinate system transformation is applied to TCS8 positions to obtain positions in "MODEL" coordinates.
250       !      During data acquisition, real time histograms will be displayed of the LDV and analog data.  After the
255       !   data have been acquired, the averages, standard deviations, and shear stresses will be calculated and displayed
260       !   in profile plots where the data are plotted versus traverse position.  The reduced data are also sent to the
265       !   printer in tabular form.  The reduced data as well as the raw data are stored along with the tunnel conditions
270       !   on the hard disc for archival purposes and also to allow for further data reduction, data plotting, and/or data
275       !   transfer to other computers.
280       !
285       ! PROGRAM OPERATION:
290       !
295       !   The following power up sequences should be completed before this program is run:
300       !           1. Turn on the "MDS" Motor Drive System boxes.
305       !           2. Turn on the "TCS8" Traverse Control System.
310       !           3. Turn on the "LVDAS" Laser Velocimeter Data Acquisition System.
315       !           4. Turn on the HP series 9000 model 375 computer.
320       !   This program will automatically be loaded and executed when the computer is turned on.  If it is not loaded,
325       !   then you can type in the following commands to load and then execute it.
330       !           LOAD "3.5'HWT:,1400,0,0"
335       !           RUN
340       !   When the program is ready for user operation, it will display three things on the CRT.  These are the main
345       !   menu, TCS8 traverse positions, and new sets of histogram and profile graphs.  If they do not appear on the CRT
350       !   then the following actions should be performed to reinitialize the systems.
355       !           1. Press shift reset on the HP series 9000 model 375 computer's keyboard.
360       !           2. Press reset on the back of the TCS8.
365       !           3. Press reset on the front (or back) of the LVDAS.
370       !           4. LOAD "3.5'HWT:,1400,0,0"
375       !           5. RUN
380       !
385       ! PROGRAM VARIABLES:
390       !
395       ! Mass Storage Variables:
400       !
405       !     System$      Tells the program where to read/store system data related files.
410       !     Data$        Tells the program where to read/store raw and reduced data related files.
415       !     File$        File name for tunnel conditions data or raw and reduced data.
420       !
425       ! Menu Variables:
430       !
435       !     Menu$(*)     String array where each element describes its corresponding menu subroutine's function.
440       !     Menu         Used as an index to the string array Menu$(*).  Indicates which of the menus has been
445       !                  selected as the current menu.
450       !     Key          Used as an index to the string array Menu$(*).  Indicates which one of eight menu
455       !                  subroutines in the menu is to be executed.
460       !     Busy         Tells the Menu Status subprogram to display the current menu selection in red  text.
465       !     Ready        Tells the Menu Status subprogram to display the current menu selection in blue text.
470       !
475       ! Traverse Position Variables:
480       !
485       !     Tun1(*)      TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in "TUNNEL" coordinates.
490       !     Tun2(*)      TCS8  receiving   side traverse positions (X2,Y2,Z2,A2) in "TUNNEL" coordinates.
495       !     Mod1(*)      TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in "MODEL" coordinates.
```

```
!     Mod2(*)       TCSB transmitting side traverse positions (X2,Y2,Z2,A2) in "MODEL" coordinates.
!     Side$         Indicates which sides are to be moved:
!                        Tx      : Transmitting side only.
!                        Rx      : Receiving side only.
!                        Tx & Rx : Both sides together.
!     Coor$         Indicates which coordinate system the movement is to be made in:
!                        TUNNEL  : TUNNEL coordinates.
!                        MODEL   : MODEL  coordinates.
!     Mode$         Indicates which movement mode is to be completed:
!                        RELATIVE: Movements are relative to current positions.
!                        ABSOLUTE: Movements are to absolute positions.
!     Movement      Indicates the desired movement for the selected axis.
!     Paxis         Specifies which axis is to be traversed for the profile.  Also defines axis for plots.
!
! Auto Move Traverse Position Variables:
!
!     Pos(*)        Array of pre-programmed auto move positions.
!     Pname$(*)     Names for the variables in Pos(*).
!     Pimage$(*)    Image formats for the variables in Pos(*).
!     Punits$(*)    Units for the variables in Pos(*).
!     Npos          Number of pre-programmed auto move positions in Pos(*).
!     Paxis         Specifies which axis is to be traversed for the profile.  Also defines axis for plots.
!
! Traverse Positions and Velocity Coordinate System Transformation Variables:
!
!     Alpha(*)      Angles of attack, yaw, and roll.
!                        Alpha(1) : Angle of Attack.
!                        Alpha(2) : Angle of Yaw.
!                        Alpha(3) : Angle of Roll.
!     Mod2tun(*)    Coordinate system transformation matrix for converting positions & velocities from MODEL to TUNNEL.
!     Tun2mod(*)    Coordinate system transformation matrix for converting positions & velocities from TUNNEL to MODEL.
!
! Tunnel Condition Variables:
!
!     Array(*)      Array of tunnel conditions, laser parameters, graph scales, etc.
!     Name$(*)      Names for the variables in Array(*).
!     Image$(*)     Image formats for the variables in Array(*).
!     Units$(*)     Units for the variables in Array(*).
!
! Misc. Tunnel Condition Variables:
!
!     Date          Date.
!     Time          Time.
!     Run           Run Number.
!     File          File Number.
!     Mach          Mach Number.
!     Re_ft         Re/Ft (Reynolds Number per Foot).
!     Uedge         Freestream Velocity (m/s).
!     Uinf          Freestream Velocity (m/s).
!     Stemp         Stagnation Temperature (deg R).
!     Ttemp         Total Temperature (deg R).
!     Tt_mv         Total Temperature data in gained millivolts.
!     Tt_raw        Total Temperature raw data in ungained volts.
!     Gain          Gain for total temperature raw analog data in ungained volts to gained millivolts conversion.
!
! LVDAS Variables:
!
!     Table(*)      Lookup table of frequencies.
!     Atime         The maximum desired acquisition time (seconds).
!     Ctime         The maximum desired coincidence time (seconds).
!     At_exp        Exponent for inter-arrival times.
!     Ct_exp        Exponent for coincidence times.
!     Nreads        Number of desired samples.
!     Nsam          Number of acquired samples.
!     Coin(*)       Coincidence criteria.
!     Cmask         Coincidence mask for U,V,W selection.
!     Raw(*)        Array of raw data acquired from the LVDAS.
!
! Instantaneous Velocity and Voltage Variables:
!
!     Ui(*)         Read from LVDAS as the instantaneous U frequency data, then converted into U velocities.
!     Vi(*)         Read from LVDAS as the instantaneous V frequency data, then converted into V velocities.
!     Wi(*)         Read from LVDAS as the instantaneous W frequency data, then converted into W velocities.
!     Ai(*)         Read from LVDAS as the instantaneous A voltage data.
!     Bi(*)         Read from LVDAS as the instantaneous B voltage data.
!     Ii(*)         Read from LVDAS as the raw inter-arrival time data, then converted into inter-arrival times.
!     Ci(*)         Read from LVDAS as the raw coincidence time data, then converted into coincidence times.
!     Valid(*)      Validation words.  Initially all ones, then some set to zero during histogram clipping.
!
! Histogram Clipping Variables:
```

```
!
!      Umin          The minimum acceptable U frequency (MHz).
!      Umax          The maximum acceptable U frequency (MHz).
!      Vmin          The minimum acceptable V frequency (MHz).
!      Vmax          The maximum acceptable V frequency (MHz).
!      Wmin          The minimum acceptable W frequency (MHz).
!      Wmax          The maximum acceptable W frequency (MHz).
!      Clip          Clip:  1 turn histogram clipping on; 0 turns it off.
!
! Frequency to Velocity Conversion Variables:
!
!      Beam_spc(*)   Beam spacing at lens.
!      Focl_len(*)   Focal length.
!      Beam_sep(*)   Beam separation angle in degrees (full angle).
!      Wave_len(*)   Wave length.
!      Frng_spc(*)   Fringe Spacings.
!      Brg_frq(*)    Bragg  Frequencies.
!      Mix_frq(*)    Mixing Frequencies.
!      Mea_sgn(*)    Measured Frequencies' Signs.
!      Brg_sgn(*)    Bragg    Frequencies' Signs.
!      Mix_sgn(*)    Mixing   Frequencies' Signs.
!
! Summation Variables:
!
!      Sumu          Summation of all of the valid Ui.
!      Sumv          Summation of all of the valid Vi.
!      Sumw          Summation of all of the valid Wi.
!      Suma          Summation of all of the valid Ai.
!      Sumb          Summation of all of the valid Bi.
!      Sumi          Summation of all of the valid Ii.
!      Sumc          Summation of all of the valid Ci.
!      Sumuu         Summation of all of the valid Ui*Ui.
!      Sumvv         Summation of all of the valid Vi*Vi.
!      Sumww         Summation of all of the valid Wi*Wi.
!      Sumaa         Summation of all of the valid Ai*Ai.
!      Sumbb         Summation of all of the valid Bi*Bi.
!      Sumii         Summation of all of the valid Ii*Ii.
!      Sumcc         Summation of all of the valid Ci*Ci.
!      Sumuv         Summation of all of the valid Ui*Vi.
!      Sumvw         Summation of all of the valid Vi*Wi.
!      Sumwu         Summation of all of the valid Wi*Ui.
!      Sumab         Summation of all of the valid Ai*Bi.
!      Sumua         Summation of all of the valid Ui*Ai.
!      Sumva         Summation of all of the valid Vi*Ai.
!      Sumwa         Summation of all of the valid Wi*Ai.
!      Suml          Number of valid samples for the above summations.
!
! Reduced Data Variables:
!
!      N             Number of valid samples acquired.
!      U             Average U frequency or velocity.
!      V             Average V frequency or velocity.
!      W             Average W frequency or velocity.
!      A             Average A voltage.
!      B             Average B voltage.
!      I             Average inter-arrival time.
!      C             Average coincidence time.
!      U1            Standard deviation for U frequency or velocity.
!      V1            Standard deviation for V frequency or velocity.
!      W1            Standard deviation for W frequency or velocity.
!      A1            Standard deviation for A voltage.
!      B1            Standard deviation for B voltage.
!      I1            Standard deviation for inter-arrival time.
!      C1            Standard deviation for coincidence time.
!      U1v1          Velocity:Velocity Shear Stress.
!      V1w1          Velocity:Velocity Shear Stress.
!      W1u1          Velocity:Velocity Shear Stress.
!      A1b1          Voltage :Voltage  Cross Correlation.
!      U1a1          Velocity:Voltage  Cross Correlation.
!      V1a1          Velocity:Voltage  Cross Correlation.
!      W1a1          Velocity:Voltage  Cross Correlation.
!
! Data Plotting Symbol Variables:
!
!      Symbols(*)    Array of Symbol arrays.  Each symbol array contains a distinct geometric symbol.
!      Symbol(*)     Array of coordinates which when connected produce a distinct geometric symbol.
!      Dot(*)        Array of coordinates which produce a dot.  The dot symbol is added to all symbols.
!      Noc           The number of coordinates in a symbol.
!      Sy            Used to index the Symbols array.
!
```

```
1300     ! Histogram and Profile Graph Variables:
1305     !
1310     !       Wndw(*)       Array containing the plots' scales.
1315     !       Vwprt(*)      Array containing the plots' CRT positions.
1320     !       Xdiv(*)       Array containing the number of X divisions for the plot's X axis.
1325     !       Ydiv(*)       Array containing the number of Y divisions for the plot's Y axis.
1330     !       Xlabel$(*)    String array containing labels for the X axis.
1335     !       Ylabel$(*)    String array containing labels for the Y axis.
1340     !       Title$(*)     String array containing labels for the plots.
1345     !       Ximage$(*)    String array containing image formats for the X axis labeling.
1350     !       Yimage$(*)    String array containing image formats for the Y axis labeling.
1355     !       Legend$(*)    String array containing labels for each symbol in a profile plot.
1360     !       G             Used as an index to the above arrays.  Specifies one of nine plots.
1365     !       Gsave(*)      Used to save the entire graphics contents of the CRT.
1370     !
1375             ! Dimension the variables and arrays defined above.
1380             OPTION BASE 1
1385             COM /Pos/ Pname$(25,1)[10],Pimage$(25,1)[10],Punits$(25,1)[10],REAL Pos(25,1),Npos
1390             COM /Array/ Name$(100,4)[10],Image$(100,4)[10],Units$(100,4)[10],REAL Array(100,4)
1395             COM /Data1/ REAL Table(0:32766),INTEGER Raw(1000,10),Valid(1000)
1400             COM /Data2/ REAL Ui(1000),Vi(1000),Wi(1000),Ai(1000),Bi(1000),Ii(1000),Ci(1000)
1405             COM /Data3/ REAL Puu(1000),Pvv(1000),Pww(1000),Paa(1000),Pbb(1000),Pii(1000),Pcc(1000)
1410             COM /Data4/ REAL Puv(1000),Pvw(1000),Pwu(1000),Pab(1000),Pua(1000),Pva(1000),Pwa(1000)
1415             COM /Graph1/ Wndw(9,4),Vwprt(9,4),Xdiv(9),Ydiv(9),Xlabel$(9)[80],Ylabel$(9)[80]
1420             COM /Graph2/ Title$(9)[80],Ximage$(9)[80],Yimage$(9)[80],Legend$(9,5)[80]
1425             COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
1430             COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
1435             COM /Sum1/ REAL Sumu,Sumv,Sumw,Suma,Sumb,Sumi,Sumc,Suml
1440             COM /Sum2/ REAL Sumuu,Sumvv,Sumww,Sumaa,Sumbb,Sumii,Sumcc
1445             COM /Sum3/ REAL Sumuv,Sumvw,Sumwu,Sumab,Sumua,Sumva,Sumwu
1450             COM /Reduced/ N,U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1
1455             COM Run,File,Paxis
1460             DIM Menu$(6,8)[80],System$[20],Data$[20],File$[50],L$[160],Kbd$[160]
1465             INTEGER Gsave(1280,1024),At_exp,Ct_exp,Cmask,Nsam
1470             REAL Atime,Ctime,Symbols(5,0:20,3)
1475             DIM Tun2mod(3,3),Mod2tun(3,3),Tun1(4),Tun2(4),Mod1(4),Mod2(4),Alpha(3)
1480             DIM Beam_spc(3),Focl_len(3),Mea_sgn(3),Mix_frq(3),Mix_sgn(3),Frng_spc(3)
1485             DIM Beam_sep(3),Wave_len(3),Brg_frq(3),Brg_sgn(3),Coin(3)
1490             DEG   ! Perform trigonometric operations in degrees.
1495             !
1500             ! Perform any necessary setup and initialization routines.
1505             CALL Crt_init            ! Clear the CRT and direct printed output to it.
1510             GOSUB Lvds_set_up        ! Initialize the HP to LVDAS interface.
1515             GOSUB File_set_up        ! Select mass storage devices for system and data files.
1520             GOSUB Tcs8_set_up        ! Initialize the HP to TCS8 interface.
1525             GOSUB Menu_set_up        ! Initialize the user driven menus and display the main menu.
1530             GOSUB Grph_set_up        ! Initialize the CRT and plot the nine empty plots for profiles and histograms.
1535             !
1540 Here:       ! The main program, while continually displaying the time of day, will wait here for menu key selection.
1545             Date=TIMEDATE
1550             Time=Date
1555             PRINT PEN Blue
1560             DISP CHR$(129);"  ";TIME$(TIMEDATE);"  ";DATE$(TIMEDATE);"  ";CHR$(128)
1565             GOTO Here
1570             STOP
1575 On_key:     ON KEY 1 GOSUB Key1   ! If the user function key #1 is ever pressed then execute the "Key1" subroutine.
1580             ON KEY 2 GOSUB Key2   ! If the user function key #2 is ever pressed then execute the "Key2" subroutine.
1585             ON KEY 3 GOSUB Key3   ! If the user function key #3 is ever pressed then execute the "Key3" subroutine.
1590             ON KEY 4 GOSUB Key4   ! If the user function key #4 is ever pressed then execute the "Key4" subroutine.
1595             ON KEY 5 GOSUB Key5   ! If the user function key #5 is ever pressed then execute the "Key5" subroutine.
1600             ON KEY 6 GOSUB Key6   ! If the user function key #6 is ever pressed then execute the "Key6" subroutine.
1605             ON KEY 7 GOSUB Key7   ! If the user function key #7 is ever pressed then execute the "Key7" subroutine.
1610             ON KEY 8 GOSUB Key8   ! If the user function key #8 is ever pressed then execute the "Key8" subroutine.
1615             RETURN
1620 Keys:       !  Subroutine Key1,Key2,Key3,Key4,Key5,Key6,Key7,Key8 descriptions:
1625             !        When one of the special user function keys is pressed, the main program will execute one the
1630             !     following eight subroutines.  Each of these subroutines performs essentially the same basic
1635             !     function in that it subsequently executes one of the menu subroutines.  The particular menu
1640             !     subroutine to be executed will depend on the current menu selected and the current key pressed.
1645             !        Before the selected menu subroutine is executed, the corresponding menu entry at the top of
1650             !     the CRT is redisplayed in red text.  This indicates that the menu selection has been
1655             !     acknowledged and that any resultant actions are still in progress.  When the highlighted menu
1660             !     subroutine has completed the current TCS8 traverse positions will be read and updated on the CRT
1665             !     display.  The corresponding menu entry displayed at the top of the CRT is redisplayed in blue
1670             !     text to indicate the completion of the menu subroutine.  The user can then select another special
1675             !     function key.
1680             !  Variables:
1685             !     Menu      Indicates which of the menus has been selected as the current menu.
1690             !     Key       Indicates which one of eight menu subroutines in the menu is to be executed.
1695             !     Menu$(*)  String array where each element describes its corresponding menu subroutine's function.
```

```
1700              !      Busy      Tells the Menu Status subroutine to display the current menu selection in red text.
1705              !      Ready     Tells the Menu Status subroutine to display the current menu selection in blue text.
1710 Key1:        Key=1
1715              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1720              ON Menu GOSUB M1k1,M2k1,M3k1,M4k1,M5k1,M6k1,M7k1
1725              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1730              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1735              RETURN
1740 Key2:        Key=2
1745              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1750              ON Menu GOSUB M1k2,M2k2,M3k2,M4k2,M5k2,M6k2,M7k2
1755              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1760              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1765              RETURN
1770 Key3:        Key=3
1775              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1780              ON Menu GOSUB M1k3,M2k3,M3k3,M4k3,M5k3,M6k3,M7k3
1785              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1790              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1795              RETURN
1800 Key4:        Key=4
1805              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1810              ON Menu GOSUB M1k4,M2k4,M3k4,M4k4,M5k4,M6k4,M7k4
1815              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1820              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1825              RETURN
1830 Key5:        Key=5
1835              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1840              ON Menu GOSUB M1k5,M2k5,M3k5,M4k5,M5k5,M6k5,M7k5
1845              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1850              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1855              RETURN
1860 Key6:        Key=6
1865              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1870              ON Menu GOSUB M1k6,M2k6,M3k6,M4k6,M5k6,M6k6,M7k6
1875              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1880              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1885              RETURN
1890 Key7:        Key=7
1895              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1900              ON Menu GOSUB M1k7,M2k7,M3k7,M4k7,M5k7,M6k7,M7k7
1905              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1910              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1915              RETURN
1920 Key8:        Key=8
1925              CALL Menu_status(Menu,Key,Busy,Menu$(*))
1930              ON Menu GOSUB M1k8,M2k8,M3k8,M4k8,M5k8,M6k8,M7k8
1935              CALL Menu_status(Menu,Key,Ready,Menu$(*))
1940              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
1945              RETURN
1950 Menu1:       ! Descriptions of the "Main Menu" subroutines M1K1,...,M1K8:
1955              !        The eight subroutines M1K1,...,M1K8 together implement the "Main Menu". The following will be
1960              !     displayed at the top left of the CRT display when the "Main Menu" is selected:
1965              !
1970              !              M1K1: Laser Alignment
1975              !              M1K2: Pre Run
1980              !              M1K3: Post Run (Dump Graphics)
1985              !              M1K4: Set Auto Move Positions
1990              !              M1K5: Move traverse
1995              !              M1K6: Take data
2000              !              M1K7: Auto move and take
2005              !              M1K8: Display Histograms
2010              !
2015              !        M1K1 will change the current active menu from the "Main Menu" to the "Laser Alignment Menu".
2020              !     M1K2 will change the current active menu from the "Main Menu" to the "Pre Run Menu". M1K3 will
2025              !     transfer the graphics contents of the CRT to the printer. This provides a hard copy of the profile
2030              !     plots. M1K4 has the user enter predefined traverse positions for a profile plot. M1K5 moves the
2035              !     traverse to a user selectable position. M1K6 acquires LVDAS data at the current TCS8 traverse
2040              !     position. M1K7 acquires LVDAS data at each of the pre programed TCS8 traverse positions set up by
2045              !     M1K4. M1K8 repeatedly displays five channels of real time histograms until the user presses any
2050              !     key on the keyboard.
2055              !
2060 M1k1:        ! Change the current active menu from the "Main Menu" to the "Laser Alignment Menu".
2065              Menu=2
2070              CALL Menu_disp(Menu,Menu$(*))
2075              RETURN
2080 M1k2:        ! Change the current active menu from the "Main Menu" to the "Pre Run Menu".
2085              Menu=3
2090              CALL Menu_disp(Menu,Menu$(*))
2095              RETURN
```

```
2100 M1k3:      ! Transfer the graphics contents of the CRT to the printer.  This provides a hard copy of the plots.
2105            KEY LABELS OFF                     ! Turn off the key labels so that they won't be printed.
2110            PRINTER IS CRT;WIDTH 132
2115            DISP ""                            ! Clear the CRT's display line so that they won't be printed.
2120            FOR L=1 TO 9                       ! Clear the CRT's menu lines so that it won't be printed.
2125                PRINT TABXY(1,L);RPT$(" ",120)
2130            NEXT L
2135            PRINTER IS PRT                     ! Direct printed output to the printer.
2140            GOSUB Print_header                 ! Print the "header" tunnel conditions.
2145            CALL Dump                          ! Dump the entire CRT's contents to the printer.
2150            PRINT USING "#,@"                  ! Move to the top of the next page on the printer.
2155            PRINTER IS CRT                     ! Direct printed output to the CRT.
2160            CALL Menu_disp(Menu,Menu$(*))      ! Redisplay the menus.
2165            RETURN
2170 M1k4:      ! Have the user enter predefined traverse positions for a profile plot.
2175            CALL Enter_value("number of traverse positions",Npos,"K")
2180            REDIM Pos(Npos,1),Pname$(Npos,1),Pimage$(Npos,1),Punits$(Npos,1)
2185            MAT Pimage$= ("M4D.4D")
2190            MAT Punits$= ("in")
2195            FOR K=1 TO Npos
2200                Pname$(K,1)="Pos#"&VAL$(K)
2205            NEXT K
2210            GSTORE Gsave(*)
2215            CALL Change("VALUES",Pos(*),Pname$(*),Pimage$(*),Punits$(*))
2220            GLOAD Gsave(*)
2225            CALL Menu_disp(Menu,Menu$(*))
2230            RETURN
2235 M1k5:      ! Moves the traverse to a user selectable position.
2240            GOSUB Read_calc_fill
2245            CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
2250            CALL Enter_value(CHR$(NUM("X")+Paxis-1),Mod1(Paxis),"K")
2255 M1k5a:     ON KBD CALL Do_nothing
2260            DISP "Moving"
2265            Movement=Mod1(Paxis)
2270            CALL Tcs8move(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*),"Tx & Rx","MODEL","ABSOLUTE",
                          Paxis,Movement)
2275            CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
2280            GOSUB Calc
2285            GOSUB Fill
2290            DISP ""
2295            OFF KBD
2300            RETURN
2305 M1k6:      ! Acquire LVDAS data at the current TCS8 traverse position.
2310            DISP "Press any key to TAKE DATA"
2315            CALL Rt_histo(@Lvdas,Symbols(*),1,Kbd$)
2320            IF POS(Kbd$,"Q") THEN RETURN
2325            Cmask=Coin(1)*1+Coin(2)*2+Coin(3)*4
2330            Nsam=MIN(Nreads,1000)
2335            Date=TIMEDATE
2340            Time=Date
2345            CALL Lvdas_take(@Lvdas,Atime,Ctime,At_exp,Ct_exp,Cmask,Nsam)
2350            IF Nsam>1 THEN
2355                GOSUB Process_data
2360                OUTPUT PRT USING "K,K";CHR$(27)&"&k2S"&CHR$(27)&"&l9D",RPT$("=",140)
2365                GOSUB Store_file
2370                File=File+1
2375            END IF
2380            RETURN
2385 M1k7:      ! Acquire LVDAS data at each of the pre programed TCS8 traverse positions set up by M1K4.
2390            Quit=0
2395            FOR J=1 TO Npos
2400                CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
2405                Mod1(Paxis)=Pos(J,1)
2410                GOSUB M1k5a
2415                GOSUB M1k6
2420                IF POS(Kbd$,"Q") THEN 2430
2425            NEXT J
2430            GOSUB On_key
2435            CALL Menu_disp(Menu,Menu$(*))
2440            RETURN
2445 M1k8:      ! Repeatedly displays five channels of real time histograms until the user presses any key on the keyboard.
2450            DISP "Press any key to return to main menu"
2455            CALL Rt_histo(@Lvdas,Symbols(*),1,Kbd$)
2460            RETURN
2465 Menu2:     !  Descriptions of the "Laser Alignment Menu" subroutines M2K1,...,M2K8:
2470            !         The eight subroutines M2K1,...,M2K8 together implement the "Laser Alignment Menu".  The
2475            !     following will be displayed at the top left of the CRT display when the "Laser Alignment Menu" is
2480            !     selected:
2485            !
2490            !             M2K1: Return to main menu
```

```
2495              !            M2K2: Sides       : Tx & Rx
2500              !            M2K3: Coordinates: MODEL
2505              !            M2K4: Mode        : ABSOLUTE
2510              !            M2K5: Move X
2515              !            M2K6: Move Y
2520              !            M2K7: Move Z
2525              !            M2K8: Move A
2530              !
2535              !         M2K1 will change the current active menu from the "Laser Alignment Menu" to the "Main Menu".
2540              !      M2K2 selects whether the transmitting, receiving, or both sides of the traverse are to be moved.
2545              !      M2K3 selects the TUNNEL or MODEL coordinate systems for traverse movements.  M2K4 specifies
2550              !      movements to be relative to the currents position or to absolute positions.  M2K5 has the user
2555              !      enter a movement for the X axis and then the movement is performed.  M2K6 has the user enter
2560              !      a movement for the Y axis and then the movement is performed.  M2K7 has the user enter a movement
2565              !      for the Z axis and then the movement is performed.  M2K8 has the user enter a movement for the A
2570              !      axis and then the movement is performed.
2575              !
2580 M2k1:        ! Change the current active menu from the "Laser Alignment Menu" to the "Main Menu".
2585              Menu=1
2590              CALL Menu_disp(Menu,Menu$(*))
2595              RETURN
2600 M2k2:        ! Select whether the transmitting, receiving, or both sides of the traverse are to be moved.
2605              SELECT TRIM$(Menu$(Menu,Key)[20])
2610              CASE "Tx & Rx"
2615                  Menu$(Menu,Key)[20]="Tx"
2620              CASE "Tx"
2625                  Menu$(Menu,Key)[20]="Rx"
2630              CASE "Rx"
2635                  Menu$(Menu,Key)[20]="Tx & Rx"
2640              END SELECT
2645              CALL Menu_disp(Menu,Menu$(*))
2650              RETURN
2655 M2k3:        ! Selects the TUNNEL or MODEL coordinate systems for traverse movements.
2660              SELECT TRIM$(Menu$(Menu,Key)[20])
2665              CASE "MODEL"
2670                  Menu$(Menu,Key)[20]="TUNNEL"
2675              CASE "TUNNEL"
2680                  Menu$(Menu,Key)[20]="MODEL"
2685              END SELECT
2690              CALL Menu_disp(Menu,Menu$(*))
2695              RETURN
2700 M2k4:        ! Specifies movements to be relative to the currents position or to absolute positions.
2705              SELECT TRIM$(Menu$(Menu,Key)[20])
2710              CASE "ABSOLUTE"
2715                  Menu$(Menu,Key)[20]="RELATIVE"
2720              CASE "RELATIVE"
2725                  Menu$(Menu,Key)[20]="ABSOLUTE"
2730              END SELECT
2735              CALL Menu_disp(Menu,Menu$(*))
2740              RETURN
2745 M2k5:        !         The subroutines M2K5 thru M2K8 all execute the same code.  The code will have the user enter a
2750 M2k6:        !      movement for the X,Y,Z, or A depending on what the value of "Key" is.  The user specified movement
2755 M2k7:        !      for the selected axis will then be performed.
2760 M2k8:        !
2765              Side$=TRIM$(Menu$(Menu,2)[20])
2770              Coor$=TRIM$(Menu$(Menu,3)[20])
2775              Mode$=TRIM$(Menu$(Menu,4)[20])
2780              CALL Enter_value(Mode$&" Movement",Movement,"4D.5D")
2785              ON KBD CALL Do_nothing
2790              DISP "Moving"
2795              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
2800              CALL Tcs8move(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*),Side$,Coor$,Mode$,Key-4,Movement)
2805              CALL Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
2810              DISP ""
2815              OFF KBD
2820              RETURN
2825 Menu3:       ! Descriptions of the "Pre Run Menu" subroutines M3K1,...,M3K8:
2830              !         The eight subroutines M3K1,...,M3K8 together implement the "Pre Run Menu".  The following will
2835              !      be displayed at the top left of the CRT display when the "Pre Run Menu" is selected:
2840              !
2845              !            M3K1: Return to MAIN menu
2850              !            M3K2: Enter Run & File Numbers
2855              !            M3K3: Enter Number of Samples
2860              !            M3K4: Select Traverse Axis for Profile
2865              !            M3K5: Print Coordinate Transformation Matrices
2870              !            M3K6: Setup Graphics
2875              !            M3K7: Tunnel Conditions
2880              !            M3K8: Traverse
2885              !
2890              !         M3K1 will change the current active menu from the "Pre Run Menu" to the "Main Menu".  M3K2 has
```

```
2895           !      the user enter a the Run and File numbers.  A new run number should be assigned to each profile
2900           !      while a new file number is assigned to each set of data.  M3K3 has the user enter the desired
2905           !      number of samples.  M3K4 has the user select which axis to traverse in for the profiles.  M3K5
2910           !      prints the coordinate system transformation matrices for both traverse positions and velocities.
2915           !      M3K6 creates a new set of empty plots for new profiles.  M3K7 will change the current active menu
2920           !      from the "Pre Run Menu" to the "Tunnel Conditions Menu".  M3K8 will change the current active menu
2925           !      from the "Pre Run Menu" to the "Traverse Menu".
2930           !
2935 M3k1:     ! Change the current active menu from the "Pre Run Menu" to the "Main Menu".
2940           Menu=1
2945           CALL Menu_disp(Menu,Menu$(*))
2950           RETURN
2955 M3k2:     ! Have the user enter a the Run and File numbers.
2960           CALL Enter_value("Run",Run,"3D.2D")
2965           CALL Enter_value("File",File,"3D")
2970           RETURN
2975 M3k3:     ! Have the user enter the desired number of samples.
2980           CALL Enter_value("Number of Samples ",Nreads,"K")
2985           RETURN
2990 M3k4:     ! Have the user select which axis to traverse in for the profiles.
2995           CALL Enter_string("Traverse Axis for Profile ",Paxis$,"K")
3000           SELECT Paxis$
3005           CASE "X"
3010               Paxis=1
3015           CASE "Y"
3020               Paxis=2
3025           CASE "Z"
3030               Paxis=3
3035           CASE "A"
3040               Paxis=4
3045           CASE ELSE
3050               GOTO M3k4
3055           END SELECT
3060           GOSUB Fill
3065           RETURN
3070 M3k5:     ! Prints the coordinate system transformation matrices for both traverse positions and velocities.
3075           GOSUB Read_calc_fill
3080           OUTPUT PRT USING "#,2/"
3085           OUTPUT PRT USING "20X,K,/";"TRAVERSE COORDINATE TRANSFORMATION MATRICES"
3090           OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"TUNNEL to MODEL",Tun2mod(*)
3095           OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"MODEL to TUNNEL",Mod2tun(*)
3100           OUTPUT PRT USING "20X,K,/";"VELOCITY COORDINATE TRANSFORMATION MATRICES"
3105           OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"TUNNEL to MODEL",Tun2mod(*)
3110           OUTPUT PRT USING "20X,K,/,3(13X,3(8D.5D),/)";"MODEL to TUNNEL",Mod2tun(*)
3115           OUTPUT PRT USING "#,@"
3120           RETURN
3125 M3k6:     ! Display a new set of plots for new profiles.
3130           CALL Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
3135           RETURN
3140 M3k7:     ! Change the current active menu from the "Pre Run Menu" to the "Tunnel Conditions Menu".
3145           Menu=4
3150           CALL Menu_disp(Menu,Menu$(*))
3155           RETURN
3160 M3k8:     ! Change the current active menu from the "Pre Run Menu" to the "Traverse Menu".
3165           Menu=5
3170           CALL Menu_disp(Menu,Menu$(*))
3175           RETURN
3180 Menu4:    !  Descriptions of the "Tunnel Conditions Menu" subroutines M4K1,...,M4K8:
3185           !         The eight subroutines M4K1,...,M4K8 together implement the "Tunnel Conditions Menu".  The
3190           !      following will be displayed at the top left of the CRT display when the "Tunnel Conditions Menu" is
3195           !      selected:
3200           !
3205           !              M4K1: Return to PRE RUN menu
3210           !              M4K2: Load  Tunnel Conditions
3215           !              M4K3: Save  Tunnel Conditions
3220           !              M4K4: Print Tunnel Conditions
3225           !              M4K5: Enter Tunnel Condition Data
3230           !              M4K6: Enter Tunnel Condition Names
3235           !              M4K7: Enter Tunnel Condition Units
3240           !              M4K8: Enter Tunnel Condition Images
3245           !
3250           !         M4K1 will change the current active menu from the "Tunnel Conditions Menu" to the "Pre Run
3255           !      Menu".  M4K2 loads the old tunnel conditions from a file on the disk.  M4K3 saves the current
3260           !      tunnel conditions to a file on the disk.  M4K2 & M4K3 load and save default tunnel conditions from
3265           !      the file "ARRAY" on the hard disk.  The default values are not related to any particular run number.
3270           !      M4K4 sends the current tunnel conditions to the printer.  M4K5 has the user enter values for the
3275           !      tunnel condition variables.  M4K6 has the user enter names for the tunnel condition variables.
3280           !      M4K7 has the user enter units for the tunnel condition variables.  M4K8 has the user enter image
3285           !      formats for the tunnel condition variables.
3290           !
```

```
6495 Menu:          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
6500 Menu_read:     SUB Menu_read(Menu$(*))
6505                    !  Description:
6510                    !      This subprogram reads in the menu descriptors for each entry of the five menus.
6515                    !  Variables:
6520                    !      Menu     Used as an index to the string array Menu$(*).
6525                    !      Key      Used as an index to the string array Menu$(*).
6530                    !      Menu$(*) String array where each element describes its corresponding menu subroutine's function.
6535                    !      L$       String use to read in the menu descriptor from the data statements.
6540                    OPTION BASE 1
6545                    DIM L$[80]
6550                    ! Fill all of the menu entry's descriptions with "MxKx".
6555                    FOR Menu=1 TO SIZE(Menu$,1)
6560                        FOR Key=1 TO 8
6565                            Menu$(Menu,Key)="M"&VAL$(Menu)&"K"&VAL$(Key)&":"
6570                        NEXT Key
6575                    NEXT Menu
6580                    ON ERROR GOTO 6620     ! The following while loop will get error#36 when the data statements run out.
6585                    ! For each menu and key, enter the menu entry's description.
6590                    WHILE 1=1
6595                        READ L$
6600                        Menu=VAL(L$[2,2])
6605                        Key=VAL(L$[4,4])
6610                        Menu$(Menu,Key)=L$
6615                    END WHILE
6620                    SUBEXIT
6625                    DATA "M1K1: Laser Alignment"
6630                    DATA         "M2K1: Return to main menu"
6635                    DATA         "M2K2: Sides      : Tx & Rx"
6640                    DATA         "M2K3: Coordinates: MODEL"
6645                    DATA         "M2K4: Mode       : ABSOLUTE"
6650                    DATA         "M2K5: Move X"
6655                    DATA         "M2K6: Move Y"
6660                    DATA         "M2K7: Move Z"
6665                    DATA         "M2K8: Move A"
6670                    DATA "M1K2: Pre Run"
6675                    DATA         "M3K1: Return to MAIN menu"
6680                    DATA         "M3K2: Enter Run & File Numbers"
6685                    DATA         "M3K3: Enter Number of Samples"
6690                    DATA         "M3K4: Select Traverse Axis for Profile"
6695                    DATA         "M3K5: Print Coordinate Transformation Matrices"
6700                    DATA         "M3K6: Setup Graphics"
6705                    DATA         "M3K7: Tunnel Conditions"
6710                    DATA                 "M4K1: Return to PRE RUN menu"
6715                    DATA                 "M4K2: Load  Tunnel Conditions"
6720                    DATA                 "M4K3: Save  Tunnel Conditions"
6725                    DATA                 "M4K4: Print Tunnel Conditions"
6730                    DATA                 "M4K5: Enter Tunnel Condition Data"
6735                    DATA                 "M4K6: Enter Tunnel Condition Names"
6740                    DATA                 "M4K7: Enter Tunnel Condition Units"
6745                    DATA                 "M4K8: Enter Tunnel Condition Images"
6750                    DATA         "M3K8: Traverse"
6755                    DATA                "M5K1: Return to PRE RUN menu"
6760                    DATA                "M5K2: View & Set TCS8 Positions"
6765                    DATA                "M5K3: View & Set TCS8 Units"
6770                    DATA                "M5K4: View & Set TCS8 Revolution"
6775                    DATA                "M5K5: View & Set TCS8 Velocity"
6780                    DATA                "M5K6: View & Set TCS8 Acceleration"
6785                    DATA                "M5K8: Recalc & Replot"
6790                    DATA "M1K3: Post Run (Dump Graphics)"
6795                    DATA "M1K4: Set Auto Move Positions"
6800                    DATA "M1K5: Move traverse"
6805                    DATA "M1K6: Take data"
6810                    DATA "M1K7: Auto move and take"
6815                    DATA "M1K8: Display Histograms"
6820                SUBEND
6825 Menu_disp:     SUB Menu_disp(Menu,Menu$(*))
6830                    !  Description:
6835                    !      This subprogram displays the current menu at the top of the CRT.
6840                    !  Variables:
6845                    !      Menu     Used as an index to the string array Menu$(*).
6850                    !      Key      Used as an index to the string array Menu$(*).
6855                    !      Menu$(*) String array where each element describes its corresponding menu subroutine's function.
6860                    COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
6865                    COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
6870                    PRINTER IS CRT
6875                    PRINT PEN Blue                 ! Print the menu using blue text.
6880                    PRINT CHR$(128);CHR$(129);     ! Turn on inverse video.
6885                    IF Menu=0 THEN Menu=1
6890                    FOR Key=1 TO 8
```


11-18

PRECEDING PAGE BLANK NOT FILMED

```
6895                 Menu$(Menu,Key)=Menu$(Menu,Key)&RPT$(" ",50-LEN(Menu$(Menu,Key)))
6900                 PRINT TABXY(2,Key);" ";Menu$(Menu,Key)[3]
6905               NEXT Key
6910               PRINT CHR$(128);                 ! Turn off inverse video.
6915               PRINT PEN Black                  ! Set printing color to black.
6920             SUBEND
6925 Menu_status:  SUB Menu_status(Menu,Key,Pen,Menu$(*))
6930               !  Description:
6935               !      This subprogram displays the current menu selection in red or blue text.  The red text
6940               !      style indicates that the subroutine for the current menu selection is busy.  The blue text
6945               !      style indicates that the subroutine for the current menu selection is has completed.
6950               !  Variables:
6955               !      Menu     Indicates which of the menus has been selected as the current menu.
6960               !      Key      Indicates which one of eight menu subroutines in the menu is to be executed.
6965               !      Pen      Indicates Busy/Ready Status.  Busy: Pen=Red.  Ready: Pen=Blue.
6970               !      Menu$(*) String array.  Each element describes its corresponding menu subroutine's function.
6975               COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
6980               COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
6985               PRINT PEN Pen
6990               PRINTER IS CRT
6995               PRINT CHR$(128);CHR$(129);                       ! Turn on inverse video.
7000               PRINT TABXY(2,Key);" ";Menu$(Menu,Key)[3];CHR$(128)  ! Print menu selection & turn off inverse video.
7005               PRINT PEN Black                                  ! Set printing color to black.
7010               WAIT .1
7015             SUBEND
7020 Enter:      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
7025 Enter_value:  SUB Enter_value(Name$,Value,Image$)
7030               !  Description:
7035               !      This subprogram displays the current value of a variable and then has the user enter its new
7040               !      value.  The old value will be kept if the RETURN key is pressed and no data is entered.
7045               !  Variables:
7050               !      Name$      Name of the variable.
7055               !      Image$     Image format of the variable.  Used for printing the variable with a format.
7060               !      Value      Contains the initial value and then the updated value for the variable.
7065               IF Name$="Date" OR Name$="Time" THEN SUBEXIT
7070               DISP CHR$(129);                                ! Turn on inverse video.
7075               DISP USING 7080;Name$                          ! Display name and old value for the variable.
7080               IMAGE #,"Old ",K,"="
7085               IF Image$<>"" THEN DISP USING "#,"&Image$;Value
7090               IF Image$="" THEN DISP USING "#,K";Value
7095               DISP USING 7100;Name$
7100               IMAGE #,"     Enter new ",K
7105               INPUT " ? ",Value                              ! The user enters the new value here.
7110               DISP CHR$(128);                                ! Turn off inverse video.
7115             SUBEND
7120 Enter_string:  SUB Enter_string(Name$,Value$,Image$)
7125               !  Description:
7130               !      This subprogram displays the current value of a string variable and then has the user enter its
7135               !      new value.  The old value will be kept if the RETURN key is pressed and no data is entered.
7140               !  Variables:
7145               !      Name$      Name of the variable.
7150               !      Value$     Contains the initial value and then the updated value for the string variable.
7155               DISP CHR$(129);                                ! Turn on inverse video.
7160               DISP USING 7165;Name$                          ! Display name and old value for the string.
7165               IMAGE #,"Old ",K,"="
7170               DISP USING "#,"&Image$;Value$
7175               DISP USING 7180;Name$
7180               IMAGE #,"     Enter new ",K
7185               INPUT " ? ",Value$                             ! The user enters the new string value here.
7190               DISP CHR$(128);                                ! Turn off inverse video.
7195             SUBEND
7200 Array:      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
7205 Array_init:   SUB Array_init(Name$(*),Array(*),Image$(*),Units$(*))
7210               !  Description:
7215               !      This subprogram reads in default data for each of the variable's names, values, image formats,
7220               !      and units.  These variables include, but are not limited to, the tunnel conditions, laser
7225               !      parameters, graph scales, traverse positions, and coordinate system transformation matrices.
7230               !  Variables:
7235               !      Array(*)   Array of tunnel conditions, laser parameters, graph scales, etc.
7240               !      Name$(*)   Names for the variables in Array(*).
7245               !      Image$(*)  Image formats for the variables in Array(*).
7250               !      Units$(*)  Units for the variables in Array(*).
7255               !      X          Used as an index to the above arrays and string arrays.
7260               !      Y          Used as an index to the above arrays and string arrays.
7265               !      Before     Number of digits before the decimal point in the image format.
7270               !      After      Number of digits after  the decimal point in the image format.
7275               ON ERROR GOTO 7365
7280               READ Y
7285               FOR X=1 TO SIZE(Name$,2)
7290                   READ Name$(Y,X),Array(Y,X),Image$(Y,X),Units$(Y,X)
```

```
7295                  SELECT Image$(Y,X)
7300                  CASE "0"
7305                      Image$(Y,X)="9D"
7310                  CASE "1" TO "7"
7315                      After=VAL(Image$(Y,X))
7320                      Before=8-After
7325                      Image$(Y,X)=VAL$(Before)&"D."&VAL$(After)&"D"
7330                  CASE "K"
7335                  CASE "N"
7340                  CASE ELSE
7345                      Image$(Y,X)="9D"
7350                  END SELECT
7355              NEXT X
7360              GOTO 7280
7365              SUBEXIT
7370              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7375              DATA  1,  Date     ,    0,0,"" ,  Mach     ,  7.0,4,"" ,  STemp    ,    0,0,°R ,  Tt Gain ,  100,0,""
7380              DATA  2,  Time     ,    0,0,"" ,  Re/Ft    ,    0,0,/Ft,  TTemp    ,    0,0,°R ,  Alpha1   ,     0,4,°
7385              DATA  3,  Run      ,    0,2,"" ,  Uedge    ,    1,4,m/s,  Tt       ,    0,3,mv ,  Alpha2   ,     0,4,°
7390              DATA  4,  File     ,    0,0,"" ,  Uinf     ,    1,4,m/s,  Tt (raw),    0,3,v  ,  Alpha3   ,     0,4,°
7395              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7400              DATA 11,  Xtun1    ,    0,4,in ,  Xtun2    ,    0,4,in ,  Xmod1    ,    0,4,in ,  Xmod2    ,    0,4,in
7405              DATA 12,  Ytun1    ,    0,4,in ,  Ytun2    ,    0,4,in ,  Ymod1    ,    0,4,in ,  Ymod2    ,    0,4,in
7410              DATA 13,  Ztun1    ,    0,4,in ,  Ztun2    ,    0,4,in ,  Zmod1    ,    0,4,in ,  Zmod2    ,    0,4,in
7415              DATA 14,  Atun1    ,    0,4,in ,  Atun2    ,    0,4,in ,  Amod1    ,    0,4,in ,  Amod2    ,    0,4,in
7420              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7425              DATA 31,  UBeamSpc,.3125,3,in ,  VBeamSpc,.3125,3,in ,  WBeamSpc,.3125,3,in ,  ""       ,    0,0,""
7430              DATA 32,  UFoclLen,30.00,3,in ,  VFoclLen,30.00,3,in ,  WFoclLen,30.00,3,in ,  ""       ,    0,0,""
7435              DATA 33,  UBeamSep,0.000,3,°  ,  VBeamSep,0.000,3,°  ,  WBeamSep,0.000,3,°  ,  ""       ,    0,0,""
7440              DATA 34,  UWaveLen,514.5,3,nm ,  VWaveLen,488.0,3,nm ,  WWaveLen,476.5,3,nm ,  ""       ,    0,0,""
7445              DATA 35,  UFrngSpc,00.00,3,um ,  VFrngSpc,00.00,3,um ,  WFrngSpc,00.00,3,um ,  ""       ,    0,0,""
7450              DATA 36,  Ubrag   ,40.00,4,MHz,  Vbrag   ,40.00,4,MHz,  Wbrag   ,40.00,4,MHz,  ""       ,    0,0,""
7455              DATA 37,  Umix    , 0.00,4,MHz,  Vmix    , 0.00,4,MHz,  WMix    , 0.00,4,MHz,  ""       ,    0,0,""
7460              DATA 38,  UmeaSgn ,   -1,0,"" ,  VmeaSgn ,   +1,0,"" ,  WmeaSgn ,   +1,0,"" ,  ""       ,    0,0,""
7465              DATA 39,  UbrgSgn ,   +1,0,"" ,  VbrgSgn ,   -1,0,"" ,  WbrgSgn ,   -1,0,"" ,  ""       ,    0,0,""
7470              DATA 40,  UmixSgn ,   -1,0,"" ,  VmixSgn ,   +1,0,"" ,  WmixSgn ,   +1,0,"" ,  ""       ,    0,0,""
7475              DATA 41,  U coin  ,    1,0,"" ,  V coin  ,    1,0,"" ,  W coin  ,    0,0,"" ,  ""       ,    0,0,""
7480              DATA 42,  UFreqMin,    8,4,MHz,  VFreqMin,   25,4,MHz,  WFreqMin,  -99,4,MHz,  ""       ,    0,0,""
7485              DATA 43,  UFreqMax,   32,4,MHz,  VFreqMax,   55,4,MHz,  WFreqMax,   99,4,MHz,  *********X=4********
7490              !     Y    *********X=1*********   *********X=2*********   *********X=3*********
7495              DATA 51,  Nreads  , 1000,0,"" ,  Atime   ,    5,6,s  ,  ATexp   ,   12,0,"" ,  Paxis    ,    2,0,""
7500              DATA 52,  Nsam    , 1000,0,"" ,  Ctime   , 1E-2,6,s  ,  CTexp   ,    7,0,"" ,  Clip     ,    1,0,""
7505              DATA 53,  ""      ,    0,0,"" ,  ""      ,    0,0,"" ,  ""      ,    0,0,"" ,  Nose     ,  139,1,cm
7510              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7515              DATA 61,  Xmin1   , 0.00,0,"" ,  Xmax1   ,60.00,0,"" ,  Ymin1   ,    0,0,"" ,  Ymax1    ,  100,0,""
7520              DATA 62,  Xmin2   , 0.00,0,"" ,  Xmax2   ,60.00,0,"" ,  Ymin2   ,    0,0,"" ,  Ymax2    ,  100,0,""
7525              DATA 63,  Xmin3   , 0.00,0,"" ,  Xmax3   ,60.00,0,"" ,  Ymin3   ,    0,0,"" ,  Ymax3    ,  100,0,""
7530              DATA 64,  Xmin5   ,-5.00,0,"" ,  Xmax5   , 5.00,0,"" ,  Ymin5   ,    0,0,"" ,  Ymax5    , 1000,0,""
7535              DATA 65,  Xmin5   ,-5.00,0,"" ,  Xmax5   , 5.00,0,"" ,  Ymin5   ,    0,0,"" ,  Ymax5    , 1000,0,""
7540              DATA 66,  Xmin6   ,-0.50,1,"" ,  Xmax6   , 1.50,1,"" ,  Ymin6   , 0.00,2,"" ,  Ymax6    , 4.00,2,""
7545              DATA 67,  Xmin7   ,-5.00,0,"" ,  Xmax7   , 5.00,0,"" ,  Ymin7   , 0.00,2,"" ,  Ymax7    , 4.00,2,""
7550              DATA 68,  Xmin8   ,    0,1,"" ,  Xmax8   , 2000,1,"" ,  Ymin8   , 0.00,2,"" ,  Ymax8    , 4.00,2,""
7555              DATA 69,  Xmin9   ,-1.50,1,"" ,  Xmax9   , 1.50,1,"" ,  Ymin9   , 0.00,2,"" ,  Ymax9    , 4.00,2,""
7560              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7565              DATA 71,  Xmin1   ,  835,0,pxl,  Xmax1   , 1235,0,pxl,  Ymin1   ,  725,0,pxl,  Ymax1    ,  825,0,pxl
7570              DATA 72,  Xmin2   ,  835,0,pxl,  Xmax2   , 1235,0,pxl,  Ymin2   ,  585,0,pxl,  Ymax2    ,  685,0,pxl
7575              DATA 73,  Xmin3   ,  835,0,pxl,  Xmax3   , 1235,0,pxl,  Ymin3   ,  445,0,pxl,  Ymax3    ,  545,0,pxl
7580              DATA 74,  Xmin4   ,  835,0,pxl,  Xmax4   , 1235,0,pxl,  Ymin4   ,  305,0,pxl,  Ymax4    ,  405,0,pxl
7585              DATA 75,  Xmin5   ,  835,0,pxl,  Xmax5   , 1235,0,pxl,  Ymin5   ,  165,0,pxl,  Ymax5    ,  265,0,pxl
7590              DATA 76,  Xmin6   ,   75,0,pxl,  Xmax6   ,  325,0,pxl,  Ymin6   ,  525,0,pxl,  Ymax6    ,  825,0,pxl
7595              DATA 77,  Xmin7   ,  425,0,pxl,  Xmax7   ,  675,0,pxl,  Ymin7   ,  525,0,pxl,  Ymax7    ,  825,0,pxl
7600              DATA 78,  Xmin8   ,   75,0,pxl,  Xmax8   ,  325,0,pxl,  Ymin8   ,  165,0,pxl,  Ymax8    ,  465,0,pxl
7605              DATA 79,  Xmin9   ,  425,0,pxl,  Xmax9   ,  675,0,pxl,  Ymin9   ,  165,0,pxl,  Ymax9    ,  465,0,pxl
7610              !     Y    *********X=1*********   *********X=2*********   *********X=3*********   *********X=4********
7615              DATA 81,  Xdiv1   ,    6,0,"" ,  Ydiv1   ,    5,0,"" ,  Xdiv6   ,    4,0,"" ,  Ydiv6    ,    8,0,""
7620              DATA 82,  Xdiv2   ,    6,0,"" ,  Ydiv2   ,    5,0,"" ,  Xdiv7   ,   10,0,"" ,  Ydiv7    ,    8,0,""
7625              DATA 83,  Xdiv3   ,    6,0,"" ,  Ydiv3   ,    5,0,"" ,  Xdiv8   ,    8,0,"" ,  Ydiv8    ,    8,0,""
7630              DATA 84,  Xdiv4   ,   10,0,"" ,  Ydiv4   ,    5,0,"" ,  Xdiv9   ,    6,0,"" ,  Ydiv9    ,    8,0,""
7635              DATA 85,  Xdiv5   ,   10,0,"" ,  Ydiv5   ,    5,0,"" ,  ""      ,    0,0,"" ,  ""       ,    0,0,""
7640          SUBEND
7645 Array_print:  SUB Array_print(Array(*),Name$(*),Image$(*),Units$(*))
7650              ! Description:
7655              !     This subprogram prints the values of each of the variables with their names, image formats, and
7660              !     units.  These variables include, but are not limited to, the tunnel conditions, laser
7665              !     parameters, and graph scales.
7670              ! Variables:
7675              !     Array(*)   Array of tunnel conditions, laser parameters, graph scales, etc.
7680              !     Name$(*)   Names for the variables in Array(*).
7685              !     Image$(*)  Image formats for the variables in Array(*).
7690              !     Units$(*)  Units for the variables in Array(*).
```

```
7695                  !      X          Used as in index to the above arrays and string arrays.
7700                  !      Y          Used as in index to the above arrays and string arrays.
7705                  PRINT USING "#,5/"
7710                  FOR Y=1 TO SIZE(Array,1)
7715                      MAT SEARCH Array(Y,*),#LOC(<>0);L1
7720                      MAT SEARCH Name$(Y,*),#LOC(<>"");L2
7725                      IF L1+L2=0 AND L3=0 THEN 7845
7730                      L3=L1+L2
7735                      PRINT USING "#,28X"
7740                      FOR X=1 TO SIZE(Array,2)
7745                          SELECT Name$(Y,X)
7750                          CASE ""                     ! If the variable has no name, then print just blanks.
7755                              PRINT USING "#,28X"
7760                          CASE "Date"                 ! Use a special printing format for printing the date.
7765                              L$=DATE$(Array(Y,X))
7770                              L$=L$[1,2]&L$[4,6]&L$[8,11]
7775                              PRINT USING "#,8A,A,9A,X,3A,6X";TRIM$(Name$(Y,X)),"=",L$,Units$(Y,X)
7780                          CASE "Time"                 ! Use a special printing format for printing the time.
7785                              L$=" "&TIME$(Array(Y,X))
7790                              PRINT USING "#,8A,A,9A,X,3A,6X";TRIM$(Name$(Y,X)),"=",L$,Units$(Y,X)
7795                          CASE ELSE                   ! All others use a standard format.
7800                              IF Image$(Y,X)="" THEN Image$(Y,X)="9D"
7805                              ON ERROR GOTO 7820
7810                              PRINT USING "#,8A,A,"&Image$(Y,X)&",X,3A,6X";TRIM$(Name$(Y,X)),"=",Array(Y,X),Units$(Y,X)
7815                              GOTO 7830
7820                              OFF ERROR
7825                              PRINT USING "#,8A,A,K,X,3A,6X";TRIM$(Name$(Y,X)),"=",Array(Y,X),Units$(Y,X)
7830                          END SELECT
7835                      NEXT X
7840                      PRINT
7845                  NEXT Y
7850              SUBEND
7855 Change:      !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
7860 Change:      SUB Change(Type$,Array(*),Name$(*),Image$(*),Units$(*))
7865                !  Description:
7870                !        This subprogram displays on the CRT the values of each of the variables with their names,
7875                !     image formats, and units.  The user can select one of the variables and enter a new value,
7880                !     name, image format, or units.  The user selects the particular variable by using the
7885                !     left, right, up, and down cursor keys.  The selected variable will appear in inverse video.
7890                !     When it is not selected, it will appear in normal text.  When the user has selected the
7895                !     appropriate variable he should then press the "Select" key on the keyboard.  Then, depending on
7900                !     the value of Type$, he will be asked to enter a new value, name, image format, or units.  To
7905                !     exit the change variables mode press the "Escape" key.
7910                !        There are three types of data that are passed to the subprogram.  The first type of data
7915                !     include, but are not limited to, the tunnel conditions, laser parameters, and graph scales.
7920                !     With this first type the user is allowed to enter new variable values, names, image formats, and
7925                !     units.  The second type of data are the "Auto Move and Take" data.  These data are for the pre
7930                !     programed traverse positions used in a profile scan.  The third type of data are the "View and
7935                !     Set TCS8 parameters" data acquired from and then sent back to the TCS8.
7940                !  Variables:
7945                !     Array(*)   Array whose values, names, image formats, or units are to be modified.
7950                !     Name$(*)   Names for the variables in Array(*).
7955                !     Image$(*)  Image formats for the variables in Array(*).
7960                !     Units$(*)  Units for the variables in Array(*).
7965                !     Type$      Indicates which type of data is to be entered.
7970                !                       Type$="VALUES" has the user enter a new value for the selected variable.
7975                !                       Type$="NAMES"  has the user enter a new name  for the selected variable.
7980                !                       Type$="IMAGES" has the user enter a new image format for the selected variable.
7985                !                       Type$="UNITS"  has the user enter a new units for the selected variable.
7990                !     X,X1,X2    Used as in index to the above arrays and string arrays.
7995                !     Y,Y1,Y2    Used as in index to the above arrays and string arrays.
8000                GRAPHICS OFF          ! Turn off the graphics contents of the CRT.
8005                PRINTER IS CRT        ! Direct printed output to the CRT.
8010                FOR Y=1 TO SIZE(Array,1)
8015                    ! Search Array(*) for section containing variables.
8020                    FOR Y1=Y TO SIZE(Array,1)
8025                        FOR X=1 TO SIZE(Array,2)
8030                            IF Name$(Y1,X)<>"" THEN 8055
8035                        NEXT X
8040                    NEXT Y1
8045                    CLEAR SCREEN       ! If no more variables are found in Array(*), the Clear the CRT display and exit.
8050                    SUBEXIT
8055                    ! Search Array(*) for section empty of variables.
8060                    FOR Y2=Y1 TO SIZE(Array,1)
8065                        FOR X=1 TO SIZE(Array,2)
8070                            IF Name$(Y2,X)<>"" THEN 8085
8075                        NEXT X
8080                        GOTO 8090
8085                    NEXT Y2
8090                    ! Find the length of the following empty section.
```

```
8095                   FOR Y2=Y2 TO SIZE(Array,1)
8100                       FOR X=1 TO SIZE(Array,2)
8105                           IF Name$(Y2,X)<>"" THEN 8120
8110                       NEXT X
8115                   NEXT Y2
8120                   Y2=Y2-1
8125                   ! Clear the CRT and then display the section contain variables and the following empty section.
8130                   CLEAR SCREEN
8135                   CALL Display(Type$,Y1,Y2,Array(*),Name$(*),Image$(*),Units$(*))
8140                   Done=0
8145                   X=1
8150                   Y=Y1
8155                   ON KBD ALL,15 GOSUB Kbd
8160 Wait:             IF NOT Done THEN Wait     ! The program will wait hear until a key is pressed on the keyboard.
8165                   OFF KBD
8170                   CLEAR SCREEN
8175                   Y=Y2
8180               NEXT Y
8185               GRAPHICS ON ! Turn the graphic part of the CRT back on.
8190               SUBEXIT
8195 Kbd:          ! This subroutine will be called when one of the cursor, select, etc. keys is pressed.
8200               CALL Update(Type$,X,Y,Y1,Y2,Done,Array(*),Name$(*),Image$(*),Units$(*))
8205               RETURN
8210           SUBEND
8215 Display:  SUB Display(Type$,Y1,Y2,Array(*),Name$(*),Image$(*),Units$(*))
8220               !  Description:
8225               !      This subprogram displays on the CRT the values of each of variables with their names, image
8230               !      formats, and units.
8235               !  Variables:
8240               !      Array(*)   Array whose values, names, image formats, or units are to be modified.
8245               !      Name$(*)   Names for the variables in Array(*).
8250               !      Image$(*)  Image formats for the variables in Array(*).
8255               !      Units$(*)  Units for the variables in Array(*).
8260               !      Type$      Indicates which type of data is to be entered.
8265               !                     Type$="VALUES" has the user enter a new value for the selected variable.
8270               !                     Type$="NAMES"  has the user enter a new name  for the selected variable.
8275               !                     Type$="IMAGES" has the user enter a new image format for the selected variable.
8280               !                     Type$="UNITS"  has the user enter a new units for the selected variable.
8285               !      X,X1,X2    Used as in index to the above arrays and string arrays.
8290               !      Y,Y1,Y2    Used as in index to the above arrays and string arrays.
8295               FOR Y=Y1 TO Y2
8300                   FOR X=1 TO SIZE(Array,2)
8305                       CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
8310                   NEXT X
8315               NEXT Y
8320               CALL Select(Type$,1,Y1,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
8325           SUBEND
8330 Select:   SUB Select(Type$,X,Y,Y1,Y2,C,Array(*),Name$(*),Image$(*),Units$(*))
8335               !  Description:
8340               !      This subprogram displays on the CRT the value of one variable along with its names, image
8345               !      format, and units.
8350               !  Variables:
8355               !      Array(*)   Array whose values, names, image formats, or units are to be modified.
8360               !      Name$(*)   Names for the variables in Array(*)
8365               !      Image$(*)  Image formats for the variables in Array(*)
8370               !      Units$(*)  Units for the variables in Array(*)
8375               !      Type$      Indicates which type of data are to be entered.
8380               !                     Type$="VALUES" has the user enter a new value for the selected variable.
8385               !                     Type$="NAMES"  has the user enter a new name for the selected variable.
8390               !                     Type$="IMAGES" has the user enter a new image format for the selected variable.
8395               !                     Type$="UNITS"  has the user enter a new units for the selected variable.
8400               !      X          Used as in index to the above arrays and string arrays.
8405               !      Y,Y1,Y2    Used as in index to the above arrays and string arrays.
8410               PRINT CHR$(128+C);TABXY(26*X-24,15+Y-Y1+1);       ! If C=0 then normal.  If C=1 then inverse video.
8415               PRINT RPT$(" ",23);TABXY(26*X-24,15+Y-Y1+1);
8420               IF Name$(Y,X)="" AND Array(Y,X)=0 THEN 8520
8425               Img$=Image$(Y,X)
8430               Unt$=Units$(Y,X)
8435               IF Image$(Y,X)="" THEN Img$="K"
8440               IF Units$(Y,X)="" THEN Unt$="   "
8445               SELECT Type$
8450               CASE "VALUES"          ! If Type$="VALUES" then display the variable's value.
8455                   SELECT Name$(Y,X)
8460                   CASE "Date"
8465                   CASE "Time"
8470                   CASE ELSE
8475                       PRINT USING "#,10A,A,"&Img$&",X,3A";Name$(Y,X),":",Array(Y,X),Unt$
8480                   END SELECT
8485               CASE "NAMES"           ! If Type$="NAMES" then display the variable's name.
8490                   PRINT USING "#,10A,A,8A";Name$(Y,X),":",Name$(Y,X)
```

```
8495                 CASE "UNITS"           ! If Type$="UNITS" then display the variable's units.
8500                     PRINT USING "#,10A,A,8A";Name$(Y,X),":",Units$(Y,X)
8505                 CASE "IMAGES"          ! If Type$="IMAGES" then display the variable's image format.
8510                     PRINT USING "#,10A,A,8A";Name$(Y,X),":",Image$(Y,X)
8515                 END SELECT
8520                 PRINT CHR$(128);       ! Turn off inverse video printing.
8525             SUBEND
8530 Update:     SUB Update(Type$,X,Y,Y1,Y2,Done,Array(*),Name$(*),Image$(*),Units$(*))
8535                 !  Description:
8540                 !      This subprogram scrolls through the variables displayed on the CRT and has the user enter
8545                 !      updated values.  The user can select one of the variables and enter a new value, name, image
8550                 !      format, or units.  The user selects the particular variable by using the left, right, up, down
8555                 !      cursor keys.  This subprogram will only have been called after a keyboard key has been pressed.
8560                 !      If a cursor key has been pressed then the previously selected variable will be redisplayed in
8565                 !      normal text and the new selected variable will appear in inverse video text.  When the user has
8570                 !      selected the appropriate variable he will have pressed the "Select" key on the keyboard.  Then,
8575                 !      depending on the value of the Type$ he will be asked to enter a new value, name, image format,
8580                 !      or units.  To exit the change variables mode the user will have pressed the "Escape" key.
8585                 !  Variables:
8590                 !      Array(*)    Array of tunnel conditions, laser parameters, graph scales, etc.
8595                 !      Name$(*)    Names for the variables in Array(*).
8600                 !      Image$(*)   Image formats for the variables in Array(*).
8605                 !      Units$(*)   Units for the variables in Array(*).
8610                 !      Type$       Indicates which type of data is to be entered.
8615                 !                      Type$="VALUES" has the user enter a new value for the selected variable.
8620                 !                      Type$="NAMES"  has the user enter a new name  for the selected variable.
8625                 !                      Type$="IMAGES" has the user enter a new image format for the selected variable.
8630                 !                      Type$="UNITS"  has the user enter a new units for the selected variable.
8635                 !      X           Used as in index to the above arrays and string arrays.
8640                 !      Y,Y1,Y2     Used as in index to the above arrays and string arrays.
8645                 DISABLE      ! Disable the keyboard.
8650                 K$=KBD$      ! Get the key pressed from the keyboards buffer.
8655                 IF K$="" THEN 8885
8660                 SELECT NUM(K$[1,1])
8665                 CASE 27                                                                   ! ESC key pressed.
8670                     Done=1
8675                 CASE 255
8680                     CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
8685                     SELECT NUM(K$[2,2])
8690                     CASE 73,80                                                            ! Break or Stop key pressed.
8695                         PAUSE
8700                     CASE 124                                                              ! Menu
8705                         Done=1
8710                     CASE 38                                                               ! Select key pressed.
8715                         CALL Select(Type$,X,Y,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
8720                         SELECT Type$
8725                         CASE "VALUES"
8730                             IF Name$(Y,X)="" THEN CALL Enter_string("Name for "&Name$(Y,X),Name$(Y,X),"K")
8735                             IF Image$(Y,X)="" THEN CALL Enter_string("Image for "&Name$(Y,X),Image$(Y,X),"K")
8740                             CALL Enter_value(Name$(Y,X),Array(Y,X),Image$(Y,X))
8745                         CASE "NAMES"
8750                             CALL Enter_string("Name for "&Name$(Y,X),Name$(Y,X),"K")
8755                         CASE "UNITS"
8760                             CALL Enter_string("Units for "&Name$(Y,X),Units$(Y,X),"K")
8765                         CASE "IMAGES"
8770                             CALL Enter_string("Image for "&Name$(Y,X),Image$(Y,X),"K")
8775                         END SELECT
8780                         CALL Select(Type$,X,Y,Y1,Y2,0,Array(*),Name$(*),Image$(*),Units$(*))
8785                         IF X=SIZE(Array,2) THEN Y=Y+1
8790                         X=X+1
8795                     CASE 60                                                               ! Left key pressed.
8800                         X=X-1
8805                     CASE 62                                                               ! Right key pressed.
8810                         X=X+1
8815                     CASE 94                                                               ! Up key pressed.
8820                         Y=Y-1
8825                     CASE 86                                                               ! Down key pressed.
8830                         Y=Y+1
8835                     CASE 92                                                               ! First key pressed.
8840                         X=1
8845                         Y=1
8850                     END SELECT
8855                     X=(X-1) MOD SIZE(Array,2)+1
8860                     Y=(Y-Y1+1-1) MOD (Y2-Y1+1)+Y1
8865                     IF X<1 THEN X=SIZE(Array,2)
8870                     IF Y<Y1 THEN Y=Y2
8875                     CALL Select(Type$,X,Y,Y1,Y2,1,Array(*),Name$(*),Image$(*),Units$(*))
8880                 END SELECT
8885                 ENABLE
8890                 SUBEXIT
```

```
8895                 SUBEND
8900 Misc:           !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
8905 Convert2words:  SUB Convert2words(Real,INTEGER High,Low)
8910                   !  Description:
8915                   !      This subprogram converts a single real precision variable into two 16 bit words.  The initial
8920                   !      real precision variables is converted in to a 32 bit integer and then separated into high and
8925                   !      low 16 bit integers.  The most significant 16 bits will be in the "High" variable while the
8930                   !      least significant 16 bits will be placed the the "Low" variable.  The main purpose of this
8935                   !      subprogram is to provide a means to send a 32 bit integer to the LVDAS over the 16 bit high
8940                   !      speed interface.
8945                   !  Variables:
8950                   !      Real   Initial real precision value for the variable.
8955                   !      Hex$   Hex value of "Real".  String length will be 8 bytes for 32 bits.
8960                   !      High   Most  significant 16 bits of integerized "Real".
8965                   !      Low    Least significant 16 bits of integerized "Real".
8970                   Hex$=DVAL$(Real,16)
8975                   High=IVAL(Hex$[1,4],16)
8980                   Low=IVAL(Hex$[5,8],16)
8985                 SUBEND
8990 Error:          SUB Error
8995                   !  Description:
9000                   !      This subprogram will print an error message when ever a program error occurs.  The error message
9005                   !      will be displayed at the top of the CRT and also printed on the printers paper.  Such errors
9010                   !      might occur when data to be printed will not fit in the image formats.  Other errors will also
9015                   !      generate a displayed and printed error message.
9020                   BEEP
9025                   DISP ERRM$
9030                   OUTPUT PRT;ERRM$
9035                   Prt=VAL(SYSTEM$("PRINTER IS"))
9040                   PRINTER IS CRT
9045                   PRINT TABXY(95,1);ERRM$
9050                   PRINTER IS Prt
9055                   ERROR SUBEXIT
9060                 SUBEND
9065 Scale:          SUB Scale(G)
9070                   !  Description:
9075                   !      This subprogram selects one of nine histogram or profile plots.  The plot's area of the CRT is
9080                   !      selected and scaled to the appropriate scales.
9085                   OPTION BASE 1
9090                   COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
9095                   VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
9100                   WINDOW Wndw(G,1),Wndw(G,2),Wndw(G,3),Wndw(G,4)
9105                 SUBEND
9110 Table:          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
9115 Table:          SUB Table(Table(*))
9120                   !  Description:
9125                   !      This subprogram is used to create a lookup table array.  The lookup table array facilitates
9130                   !      the rapid conversion of raw encoded Macrodyne data into a usable frequency.  Once the table
9135                   !      has been filled, then the raw Macrodyne data can be used as an index to the table array.
9140                   !  Variables:
9145                   !      Table(*)      Lookup table of frequencies.
9150                   !      Mantissa(*)   The 10 bit mantissa part of the raw Macrodyne data (0..1023).
9155                   !      Fringes       The 1 bit Fringe Count part of the raw Macrodyne data (0:16, 1:8 fringes).
9160                   !      Exponent      The 4 bit Exponent part of the raw Macrodyne data.
9165                   !      Time(*)       An array of measurement times for a given number of Fringes and Exponent.
9170                   !      Freq(*)       An array of measured frequencies for a given number of Fringes and Exponent.
9175                   !      Bin           Used to index Mantissa(*).
9180                   !      Min           Used as a subrange index for Table(*).
9185                   !      Max           Used as a subrange index for Table(*).
9190                   OPTION BASE 1
9195                   REAL Mantissa(0:1023),Time(0:1023),Freq(0:1023)
9200                   ! If the last entry in the table in not zero then the table has already been created.
9205                   IF Table(32766) THEN SUBEXIT
9210                   FOR Bin=0 TO 1023             !  Fill Mantissa array.
9215                       Mantissa(Bin)=Bin
9220                   NEXT Bin
9225                   Mantissa(0)=1
9230                   Min=0
9235                   FOR Fringes=0 TO 1            !  0 indicates 16 fringes while 1 indicates 8 fringes.
9240                       FOR Exponent=0 TO 15
9245                           Max=Min+1023
9250                           IF Max=32767 THEN     !  Maximum size of an array is 32766.
9255                               Max=32766
9260                               REDIM Mantissa(0:1022),Time(0:1022),Freq(0:1022)
9265                           END IF
9270                           DISP Fringes,Exponent
9275                           MAT Time= Mantissa*(2^(Exponent-1)/500000000)          ! Use this line with new macrodynes.
9280                          !MAT Time= Mantissa*(2^(Exponent-3)/500000000)          ! Use this line with old macrodynes.
9285                           MAT Freq= (2^(4-Fringes))/Time
9290                           MAT Freq= Freq/(1000000)
```

```
9295                          MAT Table(Min:Max)= Freq
9300                          Min=Min+1024
9305                      NEXT Exponent
9310                  NEXT Fringes
9315              SUBEND
9320 Lvdas:       !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
9325 Lvdas_init:  SUB Lvdas_init(@Gpio)
9330                  !  Description:
9335                  !      This subprogram is used to initialize the HP98622-66501 Rev B 16-bit General Purpose
9340                  !    Input Output (GPIO) interface.  The subprogram also opens the LVDAS path on the HP computer
9345                  !    for command and data transfer.  The I/O path is given the name "@Lvdas".  Data transferred
9350                  !    from the HP to the LVDAS will use the "OUTPUT @Lvdas" statement.  Data transferred to the HP
9355                  !    from LVDAS will use the "ENTER @Lvdas" statement.
9360                  !      The I/O path has a select code of 12 and is initialized to perform unformatted word
9365                  !    transfers without any end of line designations.  The DIP switches on the HP98622-66501 Rev B
9370                  !    printed circuit board need to be set as shown below:
9375                  !        DIP switches for INT LVL      :  Bit1=0   Bit0=0
9380                  !        DIP switches for Select Code :  Bit4=0   Bit3=1   Bit2=1   Bit1=0   Bit0=0
9385                  !        DIP switches for DI15to08 clk:  RDY =1   BSY =0   RD  =1
9390                  !        DIP switches for DI07to00 clk:  RDY =1   BSY =0   RD  =1
9395                  !        DIP switches for Hndsk Levels:  DOUT=0   DIN =0   HSHK=1   PSTS=0   PFLG=0   PCTL=1
9400                  ASSIGN @Gpio TO 12;WORD,FORMAT OFF,EOL ""
9405                  OUTPUT @Gpio USING "#,AA";"HP"
9410              SUBEND
9415 Lvdas_take:  SUB Lvdas_take(@Lvdas,Atime,Ctime,INTEGER At_exp,Ct_exp,Cmask,Nsam)
9420                  !  Description:
9425                  !      This subprogram samples the two analog, three digital, and two external trigger channels
9430                  !      from the LVDAS.  The HP sends a "CS" to sample the LVDAS data with coincidence.  Following the
9435                  !      "CS" the HP sends the LVDAS an additional eight words to specify the acquisition and
9440                  !      coincidence times, the inter-arrival and coincidence time exponents, the coincidence mask, and
9445                  !      the number of desired samples.  After the desired number of samples is acquired or the desired
9450                  !      acquisition time expires then the LVDAS sends to the HP an updated number of samples (Nsam).
9455                  !      The updated Nsam may be less that the original Nsam if the desired acquisition time expires
9460                  !      before the desired Nsam samples are realized.
9465                  !  Variables:
9470                  !      Atime   The maximum desired acquisition time (seconds).
9475                  !      Ctime   The maximum desired coincidence time (seconds).
9480                  !      At1     The upper word of integer of 10000000*Atime.
9485                  !      At2     The lower word of integer of 10000000*Atime.
9490                  !      Ct1     The upper word of integer of 10000000*Ctime.
9495                  !      Ct2     The lower word of integer of 10000000*Ctime.
9500                  !      At_exp  Exponent for inter-arrival times.
9505                  !      Ct_exp  Exponent for coincidence times.
9510                  !      Nsam    Number of desired samples.
9515                  !      Cmask   Coincidence Mask for U,V,W selection.
9520                  !      Raw(*)  Array of raw data acquired LVDAS data.
9525                  OPTION BASE 1
9530                  COM /Data1/ REAL Table(*),INTEGER Raw(*),Valid(*)
9535                  INTEGER At1,At2,Ct1,Ct2
9540                  DISP "Taking Data"
9545                  CALL Convert2words(Atime*10000000,At1,At2)
9550                  CALL Convert2words(Ctime*10000000,Ct1,Ct2)
9555                  OUTPUT @Lvdas USING "AA,8(W)";"CS",At1,At2,Ct1,Ct2,At_exp,Ct_exp,Cmask,Nsam
9560                  ENTER @Lvdas USING "#,W";Nsam
9565                  IF Nsam=0 THEN SUBEXIT
9570                  REDIM Raw(1:Nsam,1:10)
9575                  ENTER @Lvdas USING "#,W";Raw(*)
9580              SUBEND
9585 Lvdas_sample:  SUB Lvdas_sample(@Lvdas,Channel,Table(*),REAL Vave,Vsdv,Tave,Tsdv)
9590                  !  Description:
9595                  !      This subprogram samples one of the two analog, three digital, or two external trigger channels
9600                  !      from the LVDAS.  The HP sends the "DT","SC","RM", and "ET" commands to the LVDAS.  The disable
9605                  !      timer "DT" command tells the LVDAS to disable the LVDAS's internal timer interrupts.  This
9610                  !      prevents the LVDAS front panel displays from being updated but it also ensures that the data
9615                  !      sampling will occur uninterrupted and at a maximum data rate.  The sample channel "SC" tells the
9620                  !      LVDAS to sample the specified channel and return 1000 data samples.  Inter-arrival times are
9625                  !      also returned.  The read memory "RM" command reads back the data.  The enable timer "ET"
9630                  !      command enables the LVDAS's internal timer interrupts so that the front panel displays are
9635                  !      updated.
9640                  !  Variables:
9645                  !      Channel  Specifies one of the two analog, three digital, or two external trigger channels.
9650                  !                    Channel=0:  Specifies the U digital channel.
9655                  !                    Channel=1:  Specifies the V digital channel.
9660                  !                    Channel=2:  Specifies the W digital channel.
9665                  !                    Channel=3:  Specifies the A analog channel.
9670                  !                    Channel=4:  Specifies the B analog channel.
9675                  !                    Channel=5:  Specifies the External Trigger Timer channel.
9680                  !                    Channel=6:  Specifies the Inter-arrival Timer channel.
9685                  !      Data(*)  Array of raw analog or digital data with inter-arrival time data.
9690                  !                    Data(*,1)    Upper word of inter-arrival time data.
```

```
9695              !                    Data(*,2)   Lower word of inter-arrival time data.
9700              !                    Data(*,3)   Channel number for the data sampled.
9705              !                    Data(*,4)   Data of the channel sampled.
9710              !       V(*)      Array of data for the channel sampled.
9715              !       Vv(*)     Squares of the V data array.
9720              !       T(*)      Array of inter-arrival times for the channel sampled.
9725              !       Tt(*)     Squares of the T inter-arrival time array.
9730              !       Vave      Average value of the channel's data.
9735              !       Vsdv      Standard deviation of the channel's data.
9740              !       Tave      Average value of the channel's inter-arrival time data.
9745              !       Tsdv      Standard deviation of the channel's inter-arrival time data.
9750              OPTION BASE 1
9755              INTEGER Data(1000,4),V(1000),Vv(1000),T(1000),Tt(1000)
9760              OUTPUT @Lvdas USING "#,AA";"DT"
9765              OUTPUT @Lvdas USING "#,AA,W";"SC",Channel+1        ! LVDAS expects to see 1 to 7, not 0 to 6.
9770              OUTPUT @Lvdas USING "AA";"RM"
9775              OUTPUT @Lvdas USING "W,W";IVAL("08F0",16),IVAL("0000",16)
9780              OUTPUT @Lvdas USING "W,W";IVAL("08F0",16),IVAL("1F3F",16)
9785              ENTER @Lvdas USING "#,W";Data(*)
9790              OUTPUT @Lvdas USING "#,AA";"ET"
9795              N=SIZE(Data,1)
9800              Channel=Data(1,3)
9805              SELECT Channel
9810              CASE 0,1,2                        ! Convert raw digital data to frequencies.
9815                  FOR I=1 TO N
9820                      V(I)=Table(BINAND(32767,BINCMP(Data(I,4))))
9825                  NEXT I
9830              CASE 3,4                          ! Convert raw analog data to voltages.
9835                  MAT V= Data(*,4)
9840                  MAT V= V*(5/32768)
9845              CASE 5,6                          ! The external trigger channels have no data.
9850                  MAT V= (0)
9855              END SELECT
9860              MAT Vv= V . V
9865              MAT T= Data(*,2)
9870              MAT T= T/(10000000)
9875              MAT Tt= T . T
9880              Vave=SUM(V)/N
9885              Tave=SUM(T)/N
9890              Vsdv=SQR(ABS(SUM(Vv)/N-Vave*Vave))
9895              Tsdv=SQR(ABS(SUM(Tt)/N-Tave*Tave))
9900              MAT SEARCH Data(*,1),#LOC(<>0);Bad1
9905              MAT SEARCH Data(*,2),#LOC(<0);Bad2
9910              SUBEXIT
9915              PRINT USING 9920;Channel,Vave,Vsdv,Tave,Tsdv,Bad1,Bad2
9920              IMAGE 4D,2(M8D.4D),2(M2D.6D),10X,2(5D)
9925          SUBEND
9930 Data:        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
9935 Data_reducel: SUB Data_reducel(INTEGER At_exp,Ct_exp,Nsam)
9940              !  Description:
9945              !         This subprogram separates the ten by Nsam Raw(*) data array into multiple one by Nsam
9950              !     arrays.  The frequency arrays Ui,Vi,Wi are extracted from columns 6,7,8 of the Raw data array.
9955              !     The voltage arrays Ai & Bi are extracted from columns 9 & 10 of the Raw data array.  The
9960              !     inter-arrival time array Ii is extracted from column 1 of the Raw data array.  The coincidence
9965              !     time array Ci is extracted from column 2 of the Raw data array.  The validation word array
9970              !     Valid is extracted from column 5 of the Raw data array.  If i'th sample acquired contains
9975              !     valid data, then Valid(i) will be equal to one, and zero otherwise.  All values for the Valid
9980              !     array are initially set to one by the LVDAS.
9985              !         The raw data from arrays Ui,Vi,Wi are converted into frequencies by using their initial
9990              !     values as indexes to the frequency look up table array Table(*).  The raw data from arrays
9995              !     Ai & Bi are converted into voltages by multiplying their initial values by 5 volts over 2^15.
10000             !     The raw data from array Ii are converted into inter-arrival times by multiplying their initial
10005             !     values by 2^At_exp over 10 to get us.  The raw data from array Ci are converted into
10010             !     coincidence times by multiplying their initial values by 2^Ct_exp over 10 to get us.
10015             !  Variables:
10020             !     Table(*)  Lookup table of frequencies.
10025             !     Raw(*)    Array of raw data acquired LVDAS data.
10030             !     Ui(*)     Array of extracted raw U frequency data.
10035             !     Vi(*)     Array of extracted raw V frequency data.
10040             !     Wi(*)     Array of extracted raw W frequency data.
10045             !     Ai(*)     Array of extracted raw A voltage data.
10050             !     Bi(*)     Array of extracted raw B voltage data.
10055             !     Ii(*)     Array of extracted raw inter-arrival time data.
10060             !     Ci(*)     Array of extracted raw coincidence time data.
10065             !     Valid(*)  Array of extracted raw validation words.
10070             !     At_exp    Exponent of inter-arrival times.
10075             !     Ct_exp    Exponent of coincidence times.
10080             !     Nsam      Number of samples acquired.
10085             OPTION BASE 1
10090             COM /Data1/ REAL Table(*),INTEGER Raw(*),Valid(*)
```

```
10095                  COM /Data2/ REAL Ui(*),Vi(*),Wi(*),Ai(*),Bi(*),Ii(*),Ci(*)
10100                  REDIM Ui(Nsam),Vi(Nsam),Wi(Nsam),Ai(Nsam),Bi(Nsam),Ii(Nsam),Ci(Nsam),Valid(Nsam)
10105                  DISP "Reducing Data"
10110                  MAT Valid= Raw(*,5)
10115                  MAT Ii= Raw(*,1)        ! Extract the inter-arrival times from the raw data array.
10120                  MAT Ci= Raw(*,2)        ! Extract the coincidence times from the raw data array.
10125                  MAT Ui= Raw(*,6)        ! Extract the instantaneous U velocities from the raw data array.
10130                  MAT Vi= Raw(*,7)        ! Extract the instantaneous V velocities from the raw data array.
10135                  MAT Wi= Raw(*,8)        ! Extract the instantaneous W velocities from the raw data array.
10140                  MAT Ai= Raw(*,9)        ! Extract the instantaneous A analog voltages from the raw data array.
10145                  MAT Bi= Raw(*,10)       ! Extract the instantaneous B analog voltages from the raw data array.
10150                  FOR K=1 TO Nsam
10155                     Ui(K)=Table(Ui(K))   ! The raw data of Ui is used to index the frequency lookup table.
10160                     Vi(K)=Table(Vi(K))   ! The raw data of Vi is used to index the frequency lookup table.
10165                    !Wi(K)=Table(Wi(K))   ! The raw data of Wi is used to index the frequency lookup table.
10170                  NEXT K
10175                  MAT Ai= Ai*(5/32768)    ! The raw data for Ai is converted into a voltage (+/- 5 volts).
10180                  MAT Bi= Bi*(5/32768)    ! The raw data for Bi is converted into a voltage (+/- 5 volts).
10185                  MAT Ii= Ii*(2^At_exp/10)  ! The raw data for Ii is converted into the inter-arrival time.
10190                  MAT Ci= Ci*(2^Ct_exp/10)  ! The raw data for Ci is converted into the coincidence time.
10195               SUBEND
10200 Data_reduce2: SUB Data_reduce2(Array(*))
10205                  !  Description:
10210                  !       This subprogram takes the frequency values from the arrays Ui,Vi,Wi and replaces them with
10215                  !       velocities after doing the frequency to velocity conversion.
10220                  !  Variables:
10225                  !       Array(*)     An array containing relevant LDV laser and tunnel condition parameters
10230                  !       Frng_spc(*   Fringe Spacings extracted from Array(*).
10235                  !       Brg_frq(*)   Bragg Frequencies extracted from Array(*).
10240                  !       Mix_frq(*)   Mixing Freqs. extracted from Array(*).
10245                  !       Mea_sgn(*)   Measured Freq's. Signs extracted from Array(*)
10250                  !       Brg_sgn(*)   Bragg Freq's. Signs extracted from Array(*).
10255                  !       Mix_sgn(*)   Mixing Freq's. Signs extracted from Array(*).
10260                  !       Ui(*)        Array of instantaneous U data.
10265                  !       Vi(*)        Array of instantaneous V data.
10270                  !       Wi(*)        Array of instantaneous W data.
10275                  !  Equations:
10280                  !       The following equations are used to convert the frequencies to velocities
10285                  !            Velocity = Fs * Ftotal
10290                  !            Ftotal = MeaSgn*Fmeas+BrgSgn*Fbrag+MixSgn*Fmix
10295                  OPTION BASE 1
10300                  COM /Data1/ REAL Table(0:32766),INTEGER Raw(*),Valid(*)
10305                  COM /Data2/ REAL Ui(*),Vi(*),Wi(*),Ai(*),Bi(*),Ii(*),Ci(*)
10310                  DIM Frng_spc(3),Brg_frq(3),Mix_frq(3),Mea_sgn(3),Brg_sgn(3),Mix_sgn(3)
10315                  DISP "Converting Data"
10320                  MAT Frng_spc= Array(35,1:3)
10325                  MAT Brg_frq= Array(36,1:3)
10330                  MAT Mix_frq= Array(37,1:3)
10335                  MAT Mea_sgn= Array(38,1:3)
10340                  MAT Brg_sgn= Array(39,1:3)
10345                  MAT Mix_sgn= Array(40,1:3)
10350                  MAT Ui= Ui*(Mea_sgn(1))
10355                  MAT Vi= Vi*(Mea_sgn(2))
10360                 !MAT Wi= Wi*(Mea_sgn(3))
10365                  MAT Ui= Ui+(Brg_sgn(1)*Brg_frq(1)+Mix_sgn(1)*Mix_frq(1))
10370                  MAT Vi= Vi+(Brg_sgn(2)*Brg_frq(2)+Mix_sgn(2)*Mix_frq(2))
10375                 !MAT Wi= Wi+(Brg_sgn(3)*Brg_frq(3)+Mix_sgn(3)*Mix_frq(3))
10380                  MAT Ui= Ui*(Frng_spc(1))
10385                  MAT Vi= Vi*(Frng_spc(2))
10390                 !MAT Wi= Wi*(Frng_spc(3))
10395                  MAT Wi= (0)
10400               SUBEND
10405 Data_reduce3: SUB Data_reduce3(Gain)
10410                  !  Description:
10415                  !       This subprogram takes the voltage values from the array Ai and replaces them with the total
10420                  !       temperature after doing the voltage to temperature conversion.
10425                  !  Variables:
10430                  !       Ai(*)        Array of instantaneous A data.
10435                  !       Nsam         Number of acquired samples.
10440                  !       N            Exponent for the terms in the polynomial equations.
10445                  !       An           Coefficients for the terms in the polynomial equations.
10450                  !       Gain         Gain for the analog channels voltage.
10455                  !       Mv(*)        Array of gained raw voltages converted to millivolts.
10460                  !       Mvn(*)       Array of Mv(*) values raised to the power of N.
10465                  !       Amvn(*)      Array of Mvn(*) values multiplied by the polynomial coefficients An.
10470                  !       Sum(*)       Summation of the terms of polynomial equation.
10475                  !  Equations:
10480                  !       The following equations are used to convert the voltages to temperatures.
10485                  !            Temp=A7*Ai^7 + A6*Ai^6 + ... + A0*Ai^0 + 460
10490                  DISP "Converting Data"
```

```
10495                      OPTION BASE 1
10500                      COM /Data2/ REAL Ui(*),Vi(*),Wi(*),Ai(*),Bi(*),Ii(*),Ci(*)
10505                      DIM Mv(1000),Mvn(1000),Amvn(1000),Sum(1000)
10510                      Nsam=SIZE(Ai,1)
10515                      REDIM Mv(Nsam),Mvn(Nsam),Amvn(Nsam),Sum(Nsam)
10520                      MAT Mv= Ai*(1000/Gain)         !  Tt_mv=Tt_raw/Gain*1000
10525                      MAT Sum= (0)
10530                      MAT Mvn= (1)
10535                      FOR N=0 TO 7
10540                          READ An
10545                          MAT Amvn= (An)*Mvn
10550                          MAT Sum= Sum+Amvn
10555                          MAT Mvn= Mvn . Mv
10560                      NEXT N
10565                      MAT Ai= Sum+(460)
10570                      SUBEXIT
10575                      !     A0,       A1,       A2,       A3,       A4,        A5,          A6,          A7
10580                      DATA 150,257.10163,-28.16138,6.064559,-.792687,.05708673,-.002103462,.00003110036
10585                  SUBEND
10590 Data_reduce4:    SUB Data_reduce4(Mach,Mv,Ts,Tt)
10595                      !  Description:
10600                      !      This subprogram takes the analog voltage and converts it to a temperature.
10605                      !  Variables:
10610                      !      Mach         Mach number.
10615                      !      Tt           Total Temperature in degrees Rankine.
10620                      !      Ts           Stagnation Temperature in degrees Rankine.
10625                      !      N            Exponent for the terms in the polynomial equations.
10630                      !      An           Coefficients for the terms in the polynomial equations.
10635                      !      Mv           The gained raw voltage converted to millivolts.
10640                      !  Equations:
10645                      !      The following equations are used to convert the voltages to temperatures.
10650                      !          Temp=A7*Ai^7 + A6*Ai^6 + ... + A0*Ai^0 + 460
10655                      Tt=0
10660                      FOR N=0 TO 7
10665                          READ An
10670                          Tt=Tt+An*Mv^N
10675                      NEXT N
10680                      Tt=Tt+460
10685                      Ts=.09259*Tt
10690                      IF Mach<>7 THEN BEEP
10695                      IF Mach<>7 THEN PAUSE
10700                      SUBEXIT
10705                      !     A0,       A1,       A2,       A3,       A4,        A5,          A6,          A7
10710                      DATA 150,257.10163,-28.16138,6.064559,-.792687,.05708673,-.002103462,.00003110036
10715                  SUBEND
10720 Data_clip:       SUB Data_clip(INTEGER Nsam,REAL Umin,Umax,Vmin,Vmax,Wmin,Wmax)
10725                      !  Description:
10730                      !      This subprogram compares each of the instantaneous U,V,W frequencies with user selectable
10735                      !      minimum and maximum frequencies.  If the instantaneous value is less than the desired
10740                      !      minimum, then the validation word is set to zero.  Also, if the instantaneous value is
10745                      !      greater than the desired maximum, then the validation word is set to zero.  The setting of the
10750                      !      validation words to zero will have the net effect of discarding the data samples from the data
10755                      !      set.  In other words, the data are weighted as zero for the average, sdv, shear stress, and
10760                      !      cross correlation calculations.
10765                      !  Variables:
10770                      !      Nsam      Number of samples acquired.
10775                      !      Ui(*)     Array of instantaneous U frequencies (MHz).
10780                      !      Vi(*)     Array of instantaneous V frequencies (MHz).
10785                      !      Wi(*)     Array of instantaneous W frequencies (MHz).
10790                      !      Valid(*)  Array of sample validation words.
10795                      !      Umin      The minimum acceptable U frequency (MHz).
10800                      !      Umax      The maximum acceptable U frequency (MHz).
10805                      !      Vmin      The minimum acceptable V frequency (MHz).
10810                      !      Vmax      The maximum acceptable V frequency (MHz).
10815                      !      Wmin      The minimum acceptable W frequency (MHz).
10820                      !      Wmax      The maximum acceptable W frequency (MHz).
10825                      OPTION BASE 1
10830                      COM /Data1/ REAL Table(0:32766),INTEGER Raw(*),Valid(*)
10835                      COM /Data2/ REAL Ui(*),Vi(*),Wi(*),Ai(*),Bi(*),Ii(*),Ci(*)
10840                      DISP "Clipping Histograms"
10845                      FOR K=1 TO Nsam
10850                          MAT SEARCH Ui(*),LOC(<Umin);L,K
10855                          IF L<Nsam THEN Valid(L)=0
10860                          K=L
10865                      NEXT K
10870                      FOR K=1 TO Nsam
10875                          MAT SEARCH Ui(*),LOC(>Umax);L,K
10880                          IF L<Nsam THEN Valid(L)=0
10885                          K=L
10890                      NEXT K
```

```
10895                  FOR K=1 TO Nsam
10900                      MAT SEARCH Vi(*),LOC(<Vmin);L,K
10905                      IF L<Nsam THEN Valid(L)=0
10910                      K=L
10915                  NEXT K
10920                  FOR K=1 TO Nsam
10925                      MAT SEARCH Vi(*),LOC(>Vmax);L,K
10930                      IF L<Nsam THEN Valid(L)=0
10935                      K=L
10940                  NEXT K
10945                  !FOR K=1 TO Nsam
10950                  !    MAT SEARCH Wi(*),LOC(<Wmin);L,K
10955                  !    IF L<Nsam THEN Valid(L)=0
10960                  !    K=L
10965                  !NEXT K
10970                  !FOR K=1 TO Nsam
10975                  !    MAT SEARCH Wi(*),LOC(>Wmax);L,K
10980                  !    IF L<Nsam THEN Valid(L)=0
10985                  !    K=L
10990                  !NEXT K
10995              SUBEND
11000 Data_sum:    SUB Data_sum(INTEGER Nsam)
11005                  ! Description:
11010                  !     This subprogram performs the summations on the instantaneous LDV and analog data.  Data
11015                  !     will be weighted as zero in the summations if the value of the validation word is set to zero.
11020                  !     Intermediate arrays will be made so that summations of the products of the LDV and analog data
11025                  !     can be determined.
11030                  ! Variables:
11035                  !     Nsam      Number of samples acquired.
11040                  !     Valid(*)  Array of sample validation words.
11045                  !     Ui(*)     Array of instantaneous U frequency or velocity samples.
11050                  !     Vi(*)     Array of instantaneous V frequency or velocity samples.
11055                  !     Wi(*)     Array of instantaneous W frequency or velocity samples.
11060                  !     Ai(*)     Array of instantaneous A voltage samples.
11065                  !     Bi(*)     Array of instantaneous B voltage samples.
11070                  !     Ii(*)     Array of inter-arrival times.
11075                  !     Ci(*)     Array of coincidence times.
11080                  !     Puu(*)    Instantaneous product of the instantaneous Ui & Ui.
11085                  !     Pvv(*)    Instantaneous product of the instantaneous Vi & Vi.
11090                  !     Pww(*)    Instantaneous product of the instantaneous Wi & Wi.
11095                  !     Paa(*)    Instantaneous product of the instantaneous Ai & Ai.
11100                  !     Pbb(*)    Instantaneous product of the instantaneous Bi & Bi.
11105                  !     Pii(*)    Instantaneous product of the instantaneous Ii & Ii.
11110                  !     Pcc(*)    Instantaneous product of the instantaneous Ci & Ci.
11115                  !     Puv(*)    Instantaneous product of the instantaneous Ui & Vi.
11120                  !     Pvw(*)    Instantaneous product of the instantaneous Vi & Wi.
11125                  !     Pwu(*)    Instantaneous product of the instantaneous Wi & Ui.
11130                  !     Pab(*)    Instantaneous product of the instantaneous Ai & Bi.
11135                  !     Pua(*)    Instantaneous product of the instantaneous Ui & Ai.
11140                  !     Pva(*)    Instantaneous product of the instantaneous Vi & Ai.
11145                  !     Pwa(*)    Instantaneous product of the instantaneous Wi & Ai.
11150                  !     Sumu      Summation of the array Ui.
11155                  !     Sumv      Summation of the array Vi.
11160                  !     Sumw      Summation of the array Wi.
11165                  !     Suma      Summation of the array Ai.
11170                  !     Sumb      Summation of the array Bi.
11175                  !     Sumi      Summation of the array Ii.
11180                  !     Sumc      Summation of the array Ci.
11185                  !     Sumuu     Summation of the array Puu.
11190                  !     Sumvv     Summation of the array Pvv.
11195                  !     Sumww     Summation of the array Pww.
11200                  !     Sumaa     Summation of the array Paa.
11205                  !     Sumbb     Summation of the array Pbb.
11210                  !     Sumii     Summation of the array Pii.
11215                  !     Sumcc     Summation of the array Pcc.
11220                  !     Sumuv     Summation of the array Puv.
11225                  !     Sumvw     Summation of the array Pvw.
11230                  !     Sumwu     Summation of the array Pwu.
11235                  !     Sumab     Summation of the array Pab.
11240                  !     Sumua     Summation of the array Pua.
11245                  !     Sumva     Summation of the array Pva.
11250                  !     Sumwa     Summation of the array Pwa.
11255                  !     Suml      Number of valid samples acquired.
11260                  OPTION BASE 1
11265                  COM /Data1/ REAL Table(0:32766),INTEGER Raw(*),Valid(*)
11270                  COM /Data2/ REAL Ui(*),Vi(*),Wi(*),Ai(*),Bi(*),Ii(*),Ci(*)
11275                  COM /Data3/ REAL Puu(*),Pvv(*),Pww(*),Paa(*),Pbb(*),Pii(*),Pcc(*)
11280                  COM /Data4/ REAL Puv(*),Pvw(*),Pwu(*),Pab(*),Pua(*),Pva(*),Pwa(*)
11285                  COM /Sum1/ REAL Sumu,Sumv,Sumw,Suma,Sumb,Sumi,Sumc,Suml
11290                  COM /Sum2/ REAL Sumuu,Sumvv,Sumww,Sumaa,Sumbb,Sumii,Sumcc
```

```
11295                COM /Sum3/ REAL Sumuv,Sumvw,Sumwu,Sumab,Sumua,Sumva,Sumwa
11300                REDIM Puu(Nsam),Pvv(Nsam),Pww(Nsam),Paa(Nsam),Pbb(Nsam),Pii(Nsam),Pcc(Nsam)
11305                REDIM Puv(Nsam),Pvw(Nsam),Pwu(Nsam),Pab(Nsam),Pua(Nsam),Pva(Nsam),Pwa(Nsam)
11310                DISP "Summing Data"
11315                MAT Ui= Ui . Valid           ! Ui(I) is the instantaneous velocity Ui(I).
11320                MAT Vi= Vi . Valid           ! Vi(I) is the instantaneous velocity Vi(I).
11325                MAT Wi= Wi . Valid           ! Wi(I) is the instantaneous velocity Wi(I).
11330                MAT Ai= Ai . Valid           ! Ai(I) is the instantaneous Channel #1 Analog Voltage Ai(I).
11335                MAT Bi= Bi . Valid           ! Bi(I) is the instantaneous Channel #2 Analog Voltage Bi(I).
11340                MAT Ii= Ii . Valid           ! Ii(I) is the inter-arrival time Ii(I).
11345                MAT Ci= Ci . Valid           ! Ci(I) is the coincidence time Ci(I).
11350                MAT Puu= Ui . Ui             ! Puu(I) is the square of the instantaneous velocity Ui(I).
11355                MAT Pvv= Vi . Vi             ! Pvv(I) is the square of the instantaneous velocity Vi(I).
11360                MAT Pww= Wi . Wi             ! Pww(I) is the square of the instantaneous velocity Wi(I).
11365                MAT Paa= Ai . Ai             ! Paa(I) is the square if the instantaneous Analog Voltage Ai(I).
11370                MAT Pbb= Bi . Bi             ! Pbb(I) is the square if the instantaneous Analog Voltage Bi(I).
11375                MAT Pii= Ii . Ii             ! Pii(I) is the square if the inter-arrival time Ii(I).
11380                MAT Pcc= Ci . Ci             ! Pcc(I) is the square if the coincidence time Ci(I).
11385                MAT Puv= Ui . Vi             ! Puv(I) is the product of Ui(I) and Vi(I).
11390                MAT Pvw= Vi . Wi             ! Pvw(I) is the product of Ui(I) and Wi(I).
11395                MAT Pwu= Wi . Ui             ! Pwu(I) is the product of Wi(I) and Ui(I).
11400                MAT Pab= Ai . Bi             ! Pab(I) is the product of Ai(I) and Bi(I).
11405                MAT Pua= Ui . Ai             ! Pua(I) is the product of Ui(I) and Ai(I).
11410                MAT Pva= Vi . Ai             ! Pva(I) is the product of Vi(I) and Ai(I).
11415                MAT Pwa= Wi . Ai             ! Pwa(I) is the product of Wi(I) and Ai(I).
11420                Sumu=SUM(Ui)                 ! Sumu   is the summation of Ui(I).
11425                Sumv=SUM(Vi)                 ! Sumv   is the summation of Vi(I).
11430                Sumw=SUM(Wi)                 ! Sumw   is the summation of Wi(I).
11435                Suma=SUM(Ai)                 ! Suma   is the summation of Ai(I).
11440                Sumb=SUM(Bi)                 ! Sumb   is the summation of Bi(I).
11445                Sumi=SUM(Ii)                 ! Sumi   is the summation of Ii(I).
11450                Sumc=SUM(Ci)                 ! Sumc   is the summation of Ci(I).
11455                Sumuu=SUM(Puu)               ! Sumuu  is the summation of Ui(I)*Ui(I).
11460                Sumvv=SUM(Pvv)               ! Sumvv  is the summation of Vi(I)*Vi(I).
11465                Sumww=SUM(Pww)               ! Sumww  is the summation of Wi(I)*Wi(I).
11470                Sumaa=SUM(Paa)               ! Sumaa  is the summation of Ai(I)*Ai(I).
11475                Sumbb=SUM(Pbb)               ! Sumbb  is the summation of Bi(I)*Bi(I).
11480                Sumii=SUM(Pii)               ! Sumii  is the summation of Ii(I)*Ii(I).
11485                Sumcc=SUM(Pcc)               ! Sumcc  is the summation of Ci(I)*Ci(I).
11490                Sumuv=SUM(Puv)               ! Sumuv  is the summation of Ui(I)*Vi(I).
11495                Sumvw=SUM(Pvw)               ! Sumvw  is the summation of Vi(I)*Wi(I).
11500                Sumwu=SUM(Pwu)               ! Sumwu  is the summation of Wi(I)*Ui(I).
11505                Sumab=SUM(Pab)               ! Sumab  is the summation of Ai(I)*Bi(I).
11510                Sumua=SUM(Pua)               ! Sumua  is the summation of Ui(I)*Ai(I).
11515                Sumva=SUM(Pva)               ! Sumva  is the summation of Vi(I)*Ai(I).
11520                Sumwa=SUM(Pwa)               ! Sumwa  is the summation of Wi(I)*Ai(I).
11525                Suml=SUM(Valid)              ! Suml   is the number of valid samples.
11530           SUBEND
11535 Data_calc:    SUB Data_calc
11540                ! Description:
11545                !     This subprogram uses the summations on the instantaneous LDV and analog data as well as the
11550                !     summations of the products of the LDV and analog data.  The subprogram takes these summations
11555                !     and calculates the averages, standard deviations, and shear stresses.
11560                ! Variables:
11565                !     Suml       Number of valid samples acquired.
11570                !     Sumu       Summation of the array Ui.
11575                !     Sumv       Summation of the array Vi.
11580                !     Sumw       Summation of the array Wi.
11585                !     Suma       Summation of the array Ai.
11590                !     Sumb       Summation of the array Bi.
11595                !     Sumi       Summation of the array Ii.
11600                !     Sumc       Summation of the array Ci.
11605                !     Sumuu      Summation of the array Puu.
11610                !     Sumvv      Summation of the array Pvv.
11615                !     Sumww      Summation of the array Pww.
11620                !     Sumaa      Summation of the array Paa.
11625                !     Sumbb      Summation of the array Pbb.
11630                !     Sumii      Summation of the array Pii.
11635                !     Sumcc      Summation of the array Pcc.
11640                !     Sumuv      Summation of the array Puv.
11645                !     Sumvw      Summation of the array Pvw.
11650                !     Sumwu      Summation of the array Pwu.
11655                !     Sumab      Summation of the array Pab.
11660                !     Sumua      Summation of the array Pua.
11665                !     Sumva      Summation of the array Pva.
11670                !     Sumwa      Summation of the array Pwa.
11675                !     N          Number of valid samples acquired.
11680                !     U          Average U frequency or velocity.
11685                !     V          Average V frequency or velocity.
11690                !     W          Average W frequency or velocity.
```

```
11695          !     A        Average A voltage.
11700          !     B        Average B voltage.
11705          !     I        Average inter-arrival time.
11710          !     C        Average coincidence time.
11715          !     U1       Standard deviation for U frequency or velocity.
11720          !     V1       Standard deviation for V frequency or velocity.
11725          !     W1       Standard deviation for W frequency or velocity.
11730          !     A1       Standard deviation for A voltage.
11735          !     B1       Standard deviation for B voltage.
11740          !     I1       Standard deviation for inter-arrival time.
11745          !     C1       Standard deviation for coincidence time.
11750          !     U1v1     Velocity:Velocity Shear Stress.
11755          !     V1w1     Velocity:Velocity Shear Stress.
11760          !     W1u1     Velocity:Velocity Shear Stress.
11765          !     A1b1     Voltage :Voltage  Cross Correlation.
11770          !     U1a1     Velocity:Voltage  Cross Correlation.
11775          !     V1a1     Velocity:Voltage  Cross Correlation.
11780          !     W1a1     Velocity:Voltage  Cross.Correlation.
11785          COM /Sum1/ REAL Sumu,Sumv,Sumw,Suma,Sumb,Sumi,Sumc,Sum1
11790          COM /Sum2/ REAL Sumuu,Sumvv,Sumww,Sumaa,Sumbb,Sumii,Sumcc
11795          COM /Sum3/ REAL Sumuv,Sumvw,Sumwu,Sumab,Sumua,Sumva,Sumwa
11800          COM /Reduced/ N,U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1
11805          DISP "Calculating Results"
11810          N=Sum1
11815          IF N>0 THEN
11820            U=Sumu/N
11825            V=Sumv/N
11830            W=Sumw/N
11835            A=Suma/N
11840            B=Sumb/N
11845            I=Sumi/N
11850            C=Sumc/N
11855            U1=SQR(ABS(Sumuu/N-U*U))
11860            V1=SQR(ABS(Sumvv/N-V*V))
11865            W1=SQR(ABS(Sumww/N-W*W))
11870            A1=SQR(ABS(Sumaa/N-A*A))
11875            B1=SQR(ABS(Sumbb/N-B*B))
11880            I1=SQR(ABS(Sumii/N-I*I))
11885            C1=SQR(ABS(Sumcc/N-C*C))
11890            U1v1=Sumuv/N-U*V
11895            V1w1=Sumvw/N-V*W
11900            W1u1=Sumwu/N-W*U
11905            A1b1=Sumab/N-A*B
11910            U1a1=Sumua/N-U*A
11915            V1a1=Sumva/N-V*A
11920            W1a1=Sumwa/N-W*A
11925          ELSE
11930            U=0
11935            V=0
11940            W=0
11945            A=0
11950            B=0
11955            I=0
11960            C=0
11965            U1=0
11970            V1=0
11975            W1=0
11980            A1=0
11985            B1=0
11990            I1=0
11995            C1=0
12000            U1v1=0
12005            V1w1=0
12010            W1u1=0
12015            A1b1=0
12020            U1a1=0
12025            V1a1=0
12030            W1a1=0
12035          END IF
12040        SUBEND
12045 Data_trnsfrm:  SUB Data_trnsfrm(REAL K3x3(*),U,V,W,U1,V1,W1,U1v1,V1w1,W1u1,U1a1,V1a1,W1a1)
12050          ! Description:
12055          !       This subprogram performs a coordinate system transformation on the averages, standard
12060          !    deviations, and shear stresses.  The coordinate system transformation to be applied is passed
12065          !    through the "K3X3" array.  If a TUNNEL to MODEL coordinate system transformation is to be
12070          !    performed, then the array "Tun2mod" array will be passed to the "K3X3" array.
12075          !       NOTE:  This sub-program performs a three dimensional coordinate system transformation on
12080          !    averages, standard deviations, shear stresses, and cross correlations.  It performs this
12085          !    transformation for averages, standard deviations, shear stresses, and cross correlations that
12090          !    include one or more of the velocities U,V, or W.  The delivered system is a two component
```

```
12095   !      system.  Therefore, the third component W will have been set to be equal to zero.  Other terms
12100   !      containing W are also set to zero by the main program.  (W=W1=U1w1=W1u1=W1a1=0).
12105   !  Variables:
12110   !      U         Average U velocity.
12115   !      V         Average V velocity.
12120   !      W         Average W velocity.
12125   !      U1        Standard deviation for U velocity.
12130   !      V1        Standard deviation for V velocity.
12135   !      W1        Standard deviation for W velocity.
12140   !      U1u1      Velocity:Velocity Normal Stress.
12145   !      U1v1      Velocity:Velocity Shear  Stress.
12150   !      U1w1      Velocity:Velocity Shear  Stress.
12155   !      V1u1      Velocity:Velocity Shear  Stress.
12160   !      V1v1      Velocity:Velocity Normal Stress.
12165   !      V1w1      Velocity:Velocity Shear  Stress.
12170   !      W1u1      Velocity:Velocity Shear  Stress.
12175   !      W1v1      Velocity:Velocity Shear  Stress.
12180   !      W1w1      Velocity:Velocity Normal Stress.
12185   !      U1a1      Velocity:Voltage  Cross  Correlation.
12190   !      V1a1      Velocity:Voltage  Cross  Correlation.
12195   !      W1a1      Velocity:Voltage  Cross  Correlation.
12200   !      R(*)      Original U,V,W.
12205   !      F(*)      Original U1a1,V1a1,W1a1.
12210   !      P(*)      Original stress terms U1u1,U1v1,...,W1w1.
12215   !      K3X3      Coordinate system transformation matrix for average and Velocity:Voltage cross
12220   !                correlation conversions.
12225   !      K9X9      Coordinate system transformation matrix for Velocity:Velocity normal and shear
12230   !                stress conversions.
12235   !      S(*)      Transformed U,V,W.
12240   !      H(*)      Transformed U1a1,V1a1,W1a1.
12245   !      Q(*)      Transformed stress terms U1u1,U1v1,...,W1w1.
12250   OPTION BASE 1
12255   REAL R(3),S(3),F(3),H(3),P(9),Q(9),K9x9(9,9)
12260   DISP "Transforming Results"
12265   ! Calculate U1u1,V1v1,W1w1 using U1,V1,W1.
12270   U1u1=U1*U1
12275   V1v1=V1*V1
12280   W1w1=W1*W1
12285   ! Set U1w1,V1u1,W1v1 equal to W1u1,U1v1,V1w1.
12290   U1w1=W1u1
12295   V1u1=U1v1
12300   W1v1=V1w1
12305   ! Fill the matrix R with U,V,W.
12310   R(1)=U
12315   R(2)=V
12320   R(3)=W
12325   ! Fill the matrix F with U1a1,V1a1,W1a1.
12330   F(1)=U1a1
12335   F(2)=V1a1
12340   F(3)=W1a1
12345   ! Fill the matrix P with U1u1,U1v1,U1w1,V1u1,V1v1,V1w1,W1u1,W1v1,W1w1.
12350   P(1)=U1u1
12355   P(2)=U1v1
12360   P(3)=U1w1
12365   P(4)=V1u1
12370   P(5)=V1v1
12375   P(6)=V1w1
12380   P(7)=W1u1
12385   P(8)=W1v1
12390   P(9)=W1w1
12395   ! Define the matrix K9x9 using products of the elements from then matrix K3x3.
12400   FOR X=1 TO 9
12405       FOR Y=1 TO 9
12410           Y1=((Y-1) DIV 3)+1
12415           X1=((X-1) DIV 3)+1
12420           Y2=((Y-1) MOD 3)+1
12425           X2=((X-1) MOD 3)+1
12430           K9x9(Y,X)=K3x3(Y1,X1)*K3x3(Y2,X2)
12435       NEXT Y
12440   NEXT X
12445   ! Transform matrix R to S using K3x3.
12450   MAT S= K3x3*R
12455   ! Transform matrix F to H using K3x3.
12460   MAT H= K3x3*F
12465   ! Transform matrix P to Q using K9x9.
12470   MAT Q= K9x9*P
12475   ! Extract the transformed U,V,W from the matrix S.
12480   U=S(1)
12485   V=S(2)
12490   W=S(3)
```

```
12495                 ! Extract the transformed U1a1,V1a1,W1a1 from the matrix H.
12500                 U1a1=H(1)
12505                 V1a1=H(2)
12510                 W1a1=H(3)
12515                 ! Extract the transformed U1u1,U1v1,U1w1,V1u1,V1v1,V1w1,W1u1,W1v1,W1w1 from the matrix Q.
12520                 U1u1=Q(1)
12525                 U1v1=Q(2)
12530                 U1w1=Q(3)
12535                 V1u1=Q(4)
12540                 V1v1=Q(5)
12545                 V1w1=Q(6)
12550                 W1u1=Q(7)
12555                 W1v1=Q(8)
12560                 W1w1=Q(9)
12565                 ! Calculate U1,V1,W1 using U1u1,V1v1,W1w1.
12570                 U1=SQR(ABS(U1u1))
12575                 V1=SQR(ABS(V1v1))
12580                 W1=SQR(ABS(W1w1))
12585                 ! Return transformed U,V,W,U1,V1,W1,U1v1,V1w1,W1u1,U1a1,V1a1,W1a1 to main program.
12590             SUBEND
12595 Data_print1:  SUB Data_print1(Run,File,Pos(*),C$)
12600                 ! Description:
12605                 !      This subprogram prints the averages, standard deviations, shear stresses, and cross
12610                 !      correlations in tabular form.  This subprogram prints the reduced velocity data when their
12615                 !      units are in frequency (MHz).
12620                 ! Variables:
12625                 !     U         Average U frequency (MHz).
12630                 !     V         Average V frequency (MHz).
12635                 !     W         Average W frequency (MHz).
12640                 !     A         Average A voltage.
12645                 !     B         Average B voltage.
12650                 !     I         Average inter-arrival time (us).
12655                 !     C         Average coincidence time (us).
12660                 !     U1        Standard deviation for U frequencies (MHz).
12665                 !     V1        Standard deviation for V frequencies (MHz).
12670                 !     W1        Standard deviation for W frequencies (MHz).
12675                 !     A1        Standard deviation for A voltages.
12680                 !     B1        Standard deviation for B voltages.
12685                 !     I1        Standard deviation for inter-arrival times (us).
12690                 !     C1        Standard deviation for coincidence times (us).
12695                 !     U1v1      Velocity:Velocity Shear Stress.
12700                 !     V1w1      Velocity:Velocity Shear Stress.
12705                 !     W1u1      Velocity:Velocity Shear Stress.
12710                 !     A1b1      Voltage :Voltage  Cross Correlation.
12715                 !     U1a1      Velocity:Voltage  Cross Correlation.
12720                 !     V1a1      Velocity:Voltage  Cross Correlation.
12725                 !     W1a1      Velocity:Voltage  Cross Correlation.
12730                 !     Axis      Indicates one of the three axes X,Y,Z being traversed.
12735                 !     Pos(*)    Current Traverse Positions.
12740                 !     N         Number of valid samples acquired.
12745                 !     C$        Indicates units and/or coordinate system of data printed.
12750                 OPTION BASE 1
12755                 COM /Reduced/ N,U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1
12760                 DISP "Printing Results"
12765                 ON ERROR CALL Error
12770                 PRINTER IS PRT;WIDTH 144
12775                 L$=CHR$(NUM("X")+Axis-1)&"="
12780                 PRINT USING 12820;"Xtun=",Pos(1),"in     U=",U,"MHz     U'=",U1,"MHz     U'V'=",U1v1,"
                              U'A'=",U1a1,"     CT  =",C,"us"
12785                 PRINT USING 12825;"Ytun=",Pos(2),"in     V=",V,"MHz     V'=",V1,"MHz     V'W'=",V1w1,"
                              V'A'=",V1a1,"     IAT =",I,"us"
12790                 PRINT USING 12830;"Ztun=",Pos(3),"in     W=",W,"MHz     W'=",W1,"MHz     W'U'=",W1u1,"
                              W'A'=",W1a1,"     CT' =",C1,"us"
12795                 PRINT USING 12835;"Run =",Run,"       A=",A,"v       A'=",A1,"v            ","            ","
                              IAT'=",I1,"us"
12800                 PRINT USING 12840;"File=",File,"       B=",B,"v       B'=",B1,"v       A'B'=",A1b1,"            ","
                              N   =",N,""
12805                 PRINT
12810                 PRINTER IS CRT
12815                 OFF ERROR
12820                 IMAGE          8X, K,3D.4D,    K,5D.3D,   K,5D.3D,      K,8D.2D,       K,6D.5D,      K,9D,K
12825                 IMAGE          8X, K,3D.4D,    K,5D.3D,   K,5D.3D,      K,8D.2D,       K,6D.5D,      K,9D,K
12830                 IMAGE          8X, K,3D.4D,    K,5D.3D,   K,5D.3D,      K,8D.2D,       K,6D.5D,      K,9D,K
12835                 IMAGE          8X, K,5D.2D,    K,5D.3D,   K,5D.3D,      K,11X  ,       K,12X  ,      K,9D,K
12840                 IMAGE          8X, K,5D.2D,    K,5D.3D,   K,5D.3D,      K,8D.2D,       K,12X  ,      K,9D,K
12845             SUBEND
12850 Data_print2:  SUB Data_print2(Run,File,Pos(*),C$)
12855                 ! Description:
12860                 !      This subprogram prints the averages, standard deviations, and shear stresses, and cross
12865                 !      correlations in tabular form.  This subprogram prints the reduced velocity data when their
```

```
12870                        !      units are in m/s.
12875                        !  Variables:
12880                        !      U        Average U velocity.
12885                        !      V        Average V velocity.
12890                        !      W        Average W velocity.
12895                        !      A        Average A voltage.
12900                        !      B        Average B voltage.
12905                        !      I        Average inter-arrival time.
12910                        !      C        Average coincidence time.
12915                        !      U1       Standard deviation for U velocities (m/s).
12920                        !      V1       Standard deviation for V velocities (m/s)..
12925                        !      W1       Standard deviation for W velocities (m/s)..
12930                        !      A1       Standard deviation for A voltages.
12935                        !      B1       Standard deviation for B voltages.
12940                        !      I1       Standard deviation for inter-arrival times (us).
12945                        !      C1       Standard deviation for coincidence times (us).
12950                        !      U1v1     Velocity:Velocity Shear Stress.
12955                        !      V1w1     Velocity:Velocity Shear Stress.
12960                        !      W1u1     Velocity:Velocity Shear Stress.
12965                        !      A1b1     Voltage :Voltage  Cross Correlation.
12970                        !      U1a1     Velocity:Voltage  Cross Correlation.
12975                        !      V1a1     Velocity:Voltage  Cross Correlation.
12980                        !      W1a1     Velocity:Voltage  Cross Correlation.
12985                        !      Axis     Indicates one of the three axes X,Y,Z being traversed.
12990                        !      Pos(*)   Current Traverse Positions.
12995                        !      N        Number of valid samples acquired.
13000                        !      C$       Indicates units and/or coordinate system of data printed.
13005                        OPTION BASE 1
13010                        COM /Reduced/ N,U,V,W,A,B,I,C,U1,V1,W1,A1,B1,I1,C1,U1v1,V1w1,W1u1,A1b1,U1a1,V1a1,W1a1
13015                        DISP "Printing Results"
13020                        ON ERROR CALL Error
13025                        PRINTER IS PRT;WIDTH 144
13030                        L$=CHR$(NUM("X")+Axis-1)&"="
13035                        PRINT USING 13070;"Xmod=",Pos(1),"in     U=",U,"m/s      U'=",U1,"m/s       U'V'=",U1v1,"
                                      U'A'=",U1a1,"     CT  =",C,"us"
13040                        PRINT USING 13075;"Ymod=",Pos(2),"in     V=",V,"m/s      V'=",V1,"m/s       V'W'=",V1w1,"
                                      V'A'=",V1a1,"     IAT =",I,"us"
13045                        PRINT USING 13080;"Zmod=",Pos(3),"in     W=",W,"m/s      W'=",W1,"m/s       W'U'=",W1u1,"
                                      W'A'=",W1a1,"     CT' =",C1,"us"
13050                        PRINT USING 13085;"Run =",Run,"      A=",A,"v      A'=",A1,"v           ","             ","
                                      IAT'=",I1,"us"
13055                        PRINT USING 13090;"File=",File,"      B=",B,"v      B'=",B1,"v       A'B'=",A1b1,"          ","
                                      N   =",N,""
13060                        PRINTER IS CRT
13065                        OFF ERROR
13070                        IMAGE        8X, K,3D.4D,    K,5D.3D,   K,5D.3D,     K,8D.2D,      K,6D.5D,     K,9D,K
13075                        IMAGE        8X, K,3D.4D,    K,5D.3D,   K,5D.3D,     K,8D.2D,      K,6D.5D,     K,9D,K
13080                        IMAGE        8X, K,3D.4D,    K,5D.3D,   K,5D.3D,     K,8D.2D,      K,6D.5D,     K,9D,K
13085                        IMAGE        8X, K,5D.2D,    K,5D.3D,   K,5D.3D,     K,11X  ,      K,12X  ,     K,9D,K
13090                        IMAGE        8X, K,5D.2D,    K,5D.3D,   K,5D.3D,     K,8D.2D,      K,12X  ,     K,9D,K
13095                    SUBEND
13100 Data_plot:         SUB Data_plot(Array(*),Symbols(*),Plot,Sy,Y,X)
13105                        !  Description:
13110                        !      This subprogram plots the averages, standard deviations, and shear stresses in the 4 profile
13115                        !      plots on the CRT.  This subprogram will typically be called up to 4 times for each of the
13120                        !      four profile plots.  The first profile plot will contain the average velocities and their
13125                        !      standard deviations normalized by Uedge.  The second profile plot will contain the average
13130                        !      voltages and their standard deviations for the two analog channels.  The third profile plot
13135                        !      will contain the average temperature and its standard deviation.  The forth and last profile
13140                        !      plot will contain the velocity shear stress terms.  Data points outside the plot boundaries
13145                        !      will be plotted at the plot boundary.
13150                        !  Variables:
13155                        !      Array(*)     Array containing the plot positions and scales.
13160                        !      Symbols(*)   Array of Symbol arrays.  Each symbol array contains a distinct geometric symbol.
13165                        !      Plot         Indicates which plot that the data X will be plotted against Y in.
13170                        !      Y            Vertical position of the normalized data points in the plot.
13175                        !      X            Horizontal position of the data point.
13180                        !      Wndw(*)      Array containing the plot's scales.
13185                        !      Vwprt(*)     Array containing the plot's CRT position.
13190                        !      Symbol(*)    Array containing a distinct geometric symbol.
13195                        !      Sy           Specifies which distinct geometric symbol is to be used.
13200                        !      Noc          Specifies the number of coordinates that make up the distinct geometric symbol.
13205                        OPTION BASE 1
13210                        COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
13215                        COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
13220                        DIM Wndw(4),Vwprt(4),Symbol(20,3)
13225                        DISP "Plotting Results"
13230                        MAT Wndw= Array(60+Plot,*)
13235                        MAT Vwprt= Array(70+Plot,*)
13240                        Noc=Symbols(Sy,0,1)
```

```
13245             REDIM Symbol(Noc,3)
13250             MAT Symbol= Symbols(Sy,1:Noc,*)
13255             SELECT Sy
13260             CASE 1                    ! The symbol chosen is a square with black edges and filled with red.
13265               PEN 16*Black
13270               AREA PEN 16*Red
13275             CASE 2                    ! The symbol chosen is a octagon with black edges and filled with yellow.
13280               PEN 16*Black
13285               AREA PEN 16*Yellow
13290             CASE 3                    ! The symbol chosen is a diamond with black edges and filled with green.
13295               PEN 16*Black
13300               AREA PEN 16*Green
13305             CASE 4                    ! The symbol chosen is a triangle with black edges and filled with blue.
13310               PEN 16*Black
13315               AREA PEN 16*Blue
13320             END SELECT
13325             Xm=MIN(MAX(X,Wndw(1)),Wndw(2))      ! If X is out of bounds then set X to the edge of the graph.
13330             Ym=MIN(MAX(Y,Wndw(3)),Wndw(4))      ! If Y is out of bounds then set Y to the edge of the graph.
13335             LORG 5
13340             MOVE Xm,Ym
13345             SYMBOL Symbol(*),FILL,EDGE          ! This draws the selected symbol.
13350           SUBEND
13355 Tcs8:     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
13360 Tcs8init: SUB Tcs8init(@Tcs8)
13365             ! Description:
13370             !       This subprogram is used to initialize this computer's internal RS232 serial interface.
13375             !     The subprogram also opens the TCS8 path on the Hewlett Packard series 9000 model 3XX computer
13380             !     for command and data transfer.  The I/O path is given the name "@Tcs8".  Data transferred
13385             !     from the HP to the TCS8 will use the "OUTPUT @Tcs8" statement.  Data transferred to the HP
13390             !     from TCS8 will use the "ENTER @Tcs8" statement.
13395             !       The I/O path has a select code of 9 and is initialized to perform unformatted byte
13400             !     transfers without any end of line designations.
13405             REAL I(1:8),C(1:8)
13410             ASSIGN @Tcs8 TO 9;BYTE,FORMAT OFF,EOL ""
13415             CONTROL 9,0;1                 ! Reset interface.
13420             CONTROL 9,3;9600              ! Select a baud rate of 9600.
13425             CONTROL 9,4;31                ! Select even parity, enable parity, 2 stop bits, 8 bits per character.
13430             CONTROL 9,12;IVAL("EF",16)    ! Enable Carrier Detect.  Disable Data Set Ready.  Disable Clear To Send.
13435             CONTROL 9,13;9600             ! Default baud rate of 9600.
13440             CONTROL 9,14;31               ! Default character format: Even parity enabled, 2 stop, 8 bits/ char.
13445           SUBEND
13450 Tcs8set:  SUB Tcs8set(C$,@Tcs8)
13455             ! Description:
13460             !     This subprogram allows the user to view and then set the various initialization parameters
13465             !     of each channel of the TCS8.  These parameters are the current position, counts per inch,
13470             !     counts per revolution, motor velocity, motor acceleration, plus and minus limit switches,
13475             !     home switch, and motor stall indication.  All of these parameters can be viewed and set except
13480             !     the limit and home switches and the stall indication.  They can only be viewed.
13485             ! Variables:
13490             !     Command$   A TCS8 command string which indicates which parameter we want to view & set.
13495             !     View(*)    Array of old TCS8 parameters viewed (received from TCS8).  One for each channel.
13500             !     Set(*)     Array of new TCS8 parameters to be set (sent to TCS8).  One for each channel.
13505             !     Name$(*)   String array of TCS8 parameter names.
13510             !     Image$(*)  String array of image formats.
13515             !     Units$(*)  String array of units.
13520             !     Channel    Indicates the TCS8 channel number.  Used to index the above arrays.
13525             OPTION BASE 1
13530             DIM View(8,1),Set(8,2),Name$(8,1)[10],Image$(8,1)[10],Units$(8,1)[10]
13535             OUTPUT @Tcs8 USING "K,/";"V"&C$&"0"     ! Tell the TCS8 we want to View a parameter.
13540             ENTER @Tcs8 USING "8(K)";View(*)        ! Enter the parameter specified by Command$.
13545             ! Initialize the Name$,Image$,Units$ and Set arrays.
13550             DATA X1,X2,Y1,Y2,Z1,Z2,A1,A2
13555             READ Name$(*)
13560             MAT Image$= ("6D.4D")
13565             FOR Channel=1 TO 8
13570               Set(Channel,1)=Channel
13575               SELECT C$
13580               CASE "P"    !  Command$="P" indicates we want to view the encoder Positions in inches.
13585                 Name$(Channel,1)=Name$(Channel,1)&" (pos)"
13590                 Units$(Channel,1)="in"
13595               CASE "U"    !  Command$="U" indicates we want to view the Units in counts per inch.
13600                 Name$(Channel,1)=Name$(Channel,1)&" (cpi)"
13605                 Units$(Channel,1)="cnt"
13610               CASE "R"    !  Command$="R" indicates we want to view the number counts per Revolution.
13615                 Name$(Channel,1)=Name$(Channel,1)&" (cpr)"
13620                 Units$(Channel,1)="cnt"
13625               CASE "V"    !  Command$="V" indicates we want to view the Velocity in revolution per second.
13630                 Name$(Channel,1)=Name$(Channel,1)&" (vel)"
13635                 Units$(Channel,1)="rev"
13640               CASE "A"    !  Command$="A" indicates we want to view the Acceleration in revolution per second^2.
```

```
13645                     Name$(Channel,1)=Name$(Channel,1)&" (acc)"
13650                     Units$(Channel,1)="rev"
13655                 CASE "+"     !  Command$="+" indicates we want to view the current + direction limit switches.
13660                     Name$(Channel,1)=Name$(Channel,1)&" (+LS)"
13665                     Units$(Channel,1)="   "
13670                 CASE "-"     !  Command$="-" indicates we want to view the current - direction limit switches.
13675                     Name$(Channel,1)=Name$(Channel,1)&" (-LS)"
13680                     Units$(Channel,1)="   "
13685                 CASE "S"     !  Command$="S" indicates we want to view the current motor Stall indication status.
13690                     Name$(Channel,1)=Name$(Channel,1)&" (STALL)"
13695                     Units$(Channel,1)="   "
13700                 CASE "H"     !  Command$="H" indicates we want to view the current Home limit switches.
13705                     Name$(Channel,1)=Name$(Channel,1)&" (HS)"
13710                     Units$(Channel,1)="   "
13715                 END SELECT
13720             NEXT Channel
13725             !  The "Change" subprogram allows the user to see and then change the values of the viewed parameters.
13730             CALL Change("VALUES",View(*),Name$(*),Image$(*),Units$(*))
13735             !  The "Set" parameters command is now sent to the TCS8.
13740             SELECT C$
13745             CASE "P","U","R","V","A"
13750                 MAT Set(*,2)= View(*,1)
13755                 OUTPUT @Tcs8 USING 13760;"S"&C$,Set(*)
13760                 IMAGE K,8(D,":",M6D.4D,","),/
13765             END SELECT
13770         SUBEND
13775 Tcs8read:   SUB Tcs8read(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*))
13780             !  Description:
13785             !        This subprogram reads the current TCS8 positions.  The 8 positions are read in TUNNEL
13790             !    coordinates with the units being in inches.  Four of the eight positions (X1,Y1,Z1,A1) which
13795             !    are the transmitting side traverse positions are entered into the Tun1 array.  The other four
13800             !    positions (X2,Y2,Z2,A2) which are the receiving side traverse positions are entered into the
13805             !    Tun2 array.  The Tun1 & Tun2 arrays are converted from TUNNEL to MODEL coordinates.
13810             !        The current updated positions in the two coordinate systems are printed on the top of the
13815             !    CRT.  They are also returned to the main program.  The auxiliary channels A1 & A2 are not used.
13820             !    They can be used in the future to position probes such as hot wires and pitot tubes.
13825             !  Variables:
13830             !    Tun1(*)      TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TUNNEL coordinates.
13835             !    Tun2(*)      TCS8  receiving   side traverse positions (X2,Y2,Z2,A2) in TUNNEL coordinates.
13840             !    Mod1(*)      TCS8 transmitting side traverse positions in MODEL coordinates.
13845             !    Mod2(*)      TCS8  receiving   side traverse positions in MODEL coordinates.
13850             !    Tun2mod(*)   Coordinate system transformation matrix for converting TUNNEL to MODEL.
13855             !    Mod2tun(*)   Coordinate system transformation matrix for converting MODEL to TUNNEL.
13860             COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
13865             COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
13870             OUTPUT @Tcs8 USING "K,/";"VP0"
13875             ENTER @Tcs8 USING "8(K)";Tun1(1),Tun2(1),Tun1(2),Tun2(2),Tun1(3),Tun2(3),Tun1(4),Tun2(4)
13880             REDIM Tun1(1:3),Tun2(1:3),Mod1(1:3),Mod2(1:3)
13885             MAT Mod1= Tun2mod*Tun1
13890             MAT Mod2= Tun2mod*Tun2
13895             REDIM Tun1(1:4),Tun2(1:4),Mod1(1:4),Mod2(1:4)
13900             CALL Tcs8print(Tun1(*),Tun2(*),Mod1(*),Mod2(*))
13905         SUBEND
13910 Tcs8print:  SUB Tcs8print(Tun1(*),Tun2(*),Mod1(*),Mod2(*))
13915             !  Description:
13920             !    This subprogram prints the current updated TCS8 positions at the top of the CRT.  The
13925             !    positions are printed in TUNNEL and MODEL coordinates for each side (Tx & Rx).
13930             !  Variables:
13935             !    Tun1(*)      TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TUNNEL coordinates.
13940             !    Tun2(*)      TCS8  receiving   side traverse positions (X2,Y2,Z2,A2) in TUNNEL coordinates.
13945             !    Mod1(*)      TCS8 transmitting side traverse positions in MODEL coordinates.
13950             !    Mod2(*)      TCS8  receiving   side traverse positions in MODEL coordinates.
13955             COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
13960             COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
13965             PRINT PEN Red                   ! Print the traverse positions with red text.
13970             PRINT CHR$(128);CHR$(129);      ! Print using inverse video text.
13975             PRINT TABXY(52,1);"                                          "
13980             PRINT TABXY(52,2);"       TUN1      TUN2      MOD1      MOD2  "
13985             PRINT TABXY(52,3);"                                          "
13990             PRINT TABXY(52,4);
13995             PRINT USING "#,K,4(M3D.4D),X";" X:",Tun1(1),Tun2(1),Mod1(1),Mod2(1)
14000             PRINT TABXY(52,5);
14005             PRINT USING "#,K,4(M3D.4D),X";" Y:",Tun1(2),Tun2(2),Mod1(2),Mod2(2)
14010             PRINT TABXY(52,6);
14015             PRINT USING "#,K,4(M3D.4D),X";" Z:",Tun1(3),Tun2(3),Mod1(3),Mod2(3)
14020             PRINT TABXY(52,7);
14025             PRINT USING "#,K,4(M3D.4D),X";" A:",Tun1(4),Tun2(4),Mod1(4),Mod2(4)
14030             PRINT TABXY(52,8);"                                          "
14035             PRINT CHR$(128);                ! Turn off inverse video.
14040             PRINT PEN Black                 ! Set printing color to black.
```

```
14045                  SUBEND
14050 Tcs8move:        SUB Tcs8move(@Tcs8,Tun1(*),Tun2(*),Mod1(*),Mod2(*),Tun2mod(*),Mod2tun(*),Side$,Coor$,Mode$,K,Movement)
14055                     !  Description:
14060                     !      This subprogram allows for the movement of the probe volume and collecting optics in one of
14065                     !      two coordinate systems.  The two coordinate systems implemented are the TUNNEL and the MODEL
14070                     !      coordinate systems.  Two movements modes are available.  The first movement mode makes moves
14075                     !      relative to the current position.  The second movement mode makes moves to an absolute fixed
14080                     !      position.  Both the transmitting side and receiving side traverses can be moved in tandem
14085                     !      or separately.
14090                     !  Variables:
14095                     !      Tun1(*)     TCS8 transmitting side traverse positions (X1,Y1,Z1,A1) in TUNNEL coordinates.
14100                     !      Tun2(*)     TCS8  receiving   side traverse positions (X2,Y2,Z2,A2) in TUNNEL coordinates.
14105                     !      Mod1(*)     TCS8 transmitting side traverse positions in MODEL coordinates.
14110                     !      Mod2(*)     TCS8  receiving   side traverse positions in MODEL coordinates.
14115                     !      Tun2mod(*)  Coordinate system transformation matrix for converting TUNNEL to MODEL.
14120                     !      Mod2tun(*)  Coordinate system transformation matrix for converting MODEL to TUNNEL.
14125                     !      Side$       Indicates which sides are to be moved:
14130                     !                      Tx      : Transmitting side only.
14135                     !                      Rx      : Receiving side only.
14140                     !                      Tx & Rx : Both sides together.
14145                     !      Coor$       Indicates which coordinate system the movement is to be made in:
14150                     !                      TUNNEL  : TUNNEL coordinates.
14155                     !                      MODEL   : MODEL  coordinates.
14160                     !      Mode$       Indicates which movement mode is to be completed:
14165                     !                      RELATIVE: Movements are relative to current positions.
14170                     !                      ABSOLUTE: Movements are to absolute positions.
14175                     !      K           Indicates which axis of the four axes is to be moved.
14180                     !      Movement    Indicates the desired movement for the selected axis.
14185                     !      I(*)        Array of viewed TCS8 "Initialized" parameters.
14190                     !      C(*)        Array of viewed TCS8 "Currents On" parameters.
14195                     OPTION BASE 1
14200                     DIM L$[100]
14205                     ! If all of the channels have not yet been initialized, then do so now.
14210                     REAL Move(8,2),I(8),C(8)
14215                     OUTPUT @Tcs8 USING "K,/";"VI0"
14220                     ENTER @Tcs8 USING "8(K)";I(*)
14225                     IF SUM(I)<>8 THEN OUTPUT @Tcs8 USING "K,/";"SI0"
14230                     ! If all of the channels do not have their currents turned on, then do so now.
14235                     OUTPUT @Tcs8 USING "K,/";"VC0"
14240                     ENTER @Tcs8 USING "8(K)";C(*)
14245                     IF SUM(C)<>8 THEN OUTPUT @Tcs8 USING "K,/";"SC0:1,"
14250                     ! If the movement mode is to be RELATIVE, then clear all of the previously read positions.
14255                     IF Mode$="RELATIVE" THEN
14260                         MAT Tun1= (0)
14265                         MAT Tun2= (0)
14270                         MAT Mod1= (0)
14275                         MAT Mod2= (0)
14280                     END IF
14285                     ! Set the new Tun1(*) and Tun2(*) position arrays.
14290                     SELECT Coor$
14295                     CASE "MODEL"
14300                         Mod1(K)=Movement
14305                         Mod2(K)=Movement
14310                         REDIM Tun1(1:3),Tun2(1:3),Mod1(1:3),Mod2(1:3)
14315                         IF POS(Side$,"Tx") THEN MAT Tun1= Mod2tun*Mod1
14320                         IF POS(Side$,"Rx") THEN MAT Tun2= Mod2tun*Mod2
14325                         REDIM Tun1(1:4),Tun2(1:4),Mod1(1:4),Mod2(1:4)
14330                     CASE "TUNNEL"
14335                         IF POS(Side$,"Tx") THEN Tun1(K)=Movement
14340                         IF POS(Side$,"Rx") THEN Tun2(K)=Movement
14345                     END SELECT
14350                     ! File the move array.
14355                     FOR Channel=1 TO 8
14360                         Move(Channel,1)=Channel
14365                     NEXT Channel
14370                     Move(1,2)=Tun1(1)
14375                     Move(2,2)=Tun2(1)
14380                     Move(3,2)=Tun1(2)
14385                     Move(4,2)=Tun2(2)
14390                     Move(5,2)=Tun1(3)
14395                     Move(6,2)=Tun2(3)
14400                     Move(7,2)=Tun1(4)
14405                     Move(8,2)=Tun2(4)
14410                     ! Initiate the start of the move.
14415                     IF Mode$="ABSOLUTE" THEN OUTPUT @Tcs8 USING 14425;"MA",Move(*)
14420                     IF Mode$="RELATIVE" THEN OUTPUT @Tcs8 USING 14425;"MR",Move(*)
14425                     IMAGE K,8(D,":",S2D.5D,",",),/
14430                     ! The TCS8 will return the new updated positions only after the move is complete.
14435                     ENTER @Tcs8 USING "8(K)";Tun1(1),Tun2(1),Tun1(2),Tun2(2),Tun1(3),Tun2(3),Tun1(4),Tun2(4)
14440                     ! Turn off the motor drive currents.
```

```
14445                   OUTPUT @Tcs8 USING "K,/";"SC0:0,"
14450               SUBEND
14455               !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
14460 Ctm:          SUB Ctm(Alpha(*),Tun2mod(*),Mod2tun(*))
14465                   !  Description:
14470                   !      This subprogram computes directly the MODEL to TUNNEL coordinate system transformation
14475                   !      matrix "Mod2tun(*)".  However, the desired coordinate system transformation matrix "Tun2mod" is
14480                   !      required.  It is the matrix inverse of "Mod2tun".
14485                   !  Variables:
14490                   !      Alpha(*)      Angles of attack, yaw, and roll.
14495                   !      T1(*)         Partial coordinate system transformation matrix for converting from MODEL to
14500                   !                    TUNNEL coordinates.  Takes into account a model at angle of attack.
14505                   !      T2(*)         Partial coordinate system transformation matrix for converting from MODEL to
14510                   !                    TUNNEL coordinates.  Takes into account a model at angle of yaw.
14515                   !      T3(*)         Partial coordinate system transformation matrix for converting from MODEL to
14520                   !                    TUNNEL coordinates.  Takes into account a model at angle of roll.
14525                   !      Mod2tun(*)    Coordinate system transformation matrix for converting from MODEL to TUNNEL.
14530                   !      Tun2mod(*)    Coordinate system transformation matrix for converting from TUNNEL to MODEL.
14535                   OPTION BASE 1
14540                   REAL T1(3,3),T2(3,3),T3(3,3),Temp(3,3)
14545                   ! Define 1st coordinate transformation matrix for Mod2tun.
14550                   ! Rotation in the x-y plane about the z-axis.
14555                   ! Used when model is at an angle of attack.
14560                   T1(1,1)=COS(Alpha(1))
14565                   T1(1,2)=SIN(Alpha(1))
14570                   T1(1,3)=0
14575                   T1(2,1)=-SIN(Alpha(1))
14580                   T1(2,2)=COS(Alpha(1))
14585                   T1(2,3)=0
14590                   T1(3,1)=0
14595                   T1(3,2)=0
14600                   T1(3,3)=1
14605                   ! Define 2nd coordinate transformation matrix for Mod2tun.
14610                   ! Rotation in the x-z plane about the y-axis.
14615                   ! Used when model is at an angle of yaw.
14620                   T2(1,1)=COS(Alpha(2))
14625                   T2(1,2)=0
14630                   T2(1,3)=-SIN(Alpha(2))
14635                   T2(2,1)=0
14640                   T2(2,2)=1
14645                   T2(2,3)=0
14650                   T2(3,1)=SIN(Alpha(2))
14655                   T2(3,2)=0
14660                   T2(3,3)=COS(Alpha(2))
14665                   ! Define 3rd coordinate transformation matrix for Mod2tun.
14670                   ! Rotation in the y-z plane about the x-axis.
14675                   ! Used when model is at an angle of roll.
14680                   T3(1,1)=1
14685                   T3(1,2)=0
14690                   T3(1,3)=0
14695                   T3(2,1)=0
14700                   T3(2,2)=COS(Alpha(3))
14705                   T3(2,3)=SIN(Alpha(3))
14710                   T3(3,1)=0
14715                   T3(3,2)=-SIN(Alpha(3))
14720                   T3(3,3)=COS(Alpha(3))
14725                   ! Mod2tun converts MODEL coordinates to TUNNEL coordinates.
14730                   MAT Temp= T2*T1
14735                   MAT Mod2tun= T3*Temp
14740                   ! Tun2mod converts TUNNEL coordinates to MODEL coordinates.
14745                   MAT Tun2mod= INV(Mod2tun)
14750               SUBEND
14755 Color:        !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
14760 Crt_init:     SUB Crt_init
14765                   !  Description:
14770                   !      This subprogram initializes the CRT as the plotting device and clears both the alpha
14775                   !      numerics and graphics part of the CRT.  The color map for both of the alpha numeric printing
14780                   !      plains and the graphics drawing plains are defined here.
14785                   COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
14790                   COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
14795                   CALL Color          ! Define the color maps for the alpha numeric and the graphics plains.
14800                  !CALL Map            ! Draw the color map.
14805                  !CALL Dump           ! Dump the color map to the printer.
14810                   PRINTER IS CRT      ! Select the CRT as the printing device.
14815                   PRINTALL IS CRT     ! Send ERROR and DISP messages to CRT.
14820                   KEY LABELS OFF      ! Hide the special function key labels for f1..f8.
14825                   CLEAR SCREEN        ! Clear the alpha numeric printing plains of the CRT.
14830                   GCLEAR              ! Clear the graphics drawing plains of the CRT.
14835               SUBEND
14840 Color:        SUB Color
```

```
14845                 !  Description:
14850                 !      This subprogram defines the color map for both alpha numeric printing and graphics drawing.
14855                 !      Four of eight plains are dedicated to alpha numerics to provide for sixteen colors.  The other
14860                 !      four plains are dedicated to graphics to provide for sixteen colors.
14865                 COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
14870                 COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
14875                 DIM Map(255,2)
14880                 INTEGER Gmask(1)
14885                 READ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta,White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
14890                 DATA     0,    1,  2,      3,     4,    5,    6,      7,  8,     9,  10,    11,     12,    13,     14,    15
14895                 PLOTTER IS CRT,"INTERNAL";COLOR MAP      ! Select the CRT as the plotting device.
14900                 CONTROL CRT,14;3
14905                 SET PEN 0 INTENSITY 1,1,1                ! Set pen 0 equal to clear (white).
14910                 SET PEN 1 INTENSITY 0,0,0                ! Set pen 1 equal to black.
14915                 SET PEN 8 INTENSITY 1,1,1                ! Set pen 8 equal to white.
14920                 SET PEN 14 INTENSITY 26/30,16/30,8/30    ! Set pen 14 equal to brown.
14925                 SET PEN 15 INTENSITY .6,.6,.6            ! Set pen 15 equal to gray.
14930                 GESCAPE CRT,2;Map(*)                     ! Read RGB intensity for pens 0 to 255.
14935                 Gmask(0)=IVAL("11110000",2)              ! Define graphics  write  enable mask.
14940                 Gmask(1)=IVAL("11110000",2)              ! Define graphics display enable mask.
14945                 GESCAPE CRT,7,Gmask(*)                   ! Set graphics write & display enable masks.
14950                 SET ALPHA MASK IVAL("00001111",2)        ! Set alpha  write  enable mask.
14955                 SET DISPLAY MASK IVAL("00001111",2)      ! Set alpha display enable mask.
14960                 GESCAPE CRT,4                            ! Select normal dominant writing mode.
14965                 GCLEAR                                   ! Clear the graphics screen.
14970                 CLEAR SCREEN                             ! Clear the  alpha   screen.
14975                 GRAPHICS ON                              ! Turn graphics on.
14980                 ! Copy the alpha colors to the graph colors (use the same 16 alpha colors for graph colors.)
14985                 FOR Alpha=0 TO 15
14990                     FOR Graph=0 TO 15
14995                         Pen=16*Graph+Alpha                         ! Define pen number for Alpha:Graph combination.
15000                         Color=Graph*(Alpha=0)+Alpha*(Alpha<>0)     ! Choose the color for the
15005                         IF Alpha=Graph THEN Color=Black*(Alpha>1)  !         Alpha:Graph combination.
15010                         MAT Map(Pen,*)= Map(Color,*)               ! Get the RGB intensities for the color.
15015                         SET PEN Pen INTENSITY Map(Pen,0),Map(Pen,1),Map(Pen,2)    ! Set the RGB for the pen.
15020                     NEXT Graph
15025                 NEXT Alpha
15030               ! AREA PEN White                  ! Select white for area fills.
15035               ! PEN Black                       ! Select black for line drawing and labeling.
15040                 ALPHA PEN Black                 ! Select black for printing.
15045                 KEY LABELS PEN Blue             ! Select blue for special function key labels.
15050                 PRINT PEN Black                 ! Select black for printing.
15055                 KEY LABELS OFF                  ! Hide the special function key labels for f1..f8.
15060             SUBEND
15065 Map:        SUB Map
15070                 !  Description:
15075                 !      This subprogram displays the color map on the CRT.  The sixteen colors for the alpha plains are
15080                 !      superimposed on top of the graphics plains to show the dominance interaction of alpha and
15085                 !      graphics colors being printed and drawn on top of each other.
15090                 COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
15095                 COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
15100                 VIEWPORT 25/10.23,(25+16*4*10)/10.23,210/10.23,850/10.23
15105                 WINDOW 0,16,16,0
15110                 Pen=0
15115                 FOR Alpha=0 TO 15
15120                     FOR Graph=0 TO 15
15125                         AREA PEN 16*Graph+Alpha
15130                         PEN 16*Black
15135                         MOVE Alpha,Graph
15140                         RECTANGLE 1,1;FILL,EDGE
15145                         PRINT PEN Alpha
15150                         PRINT TABXY(4+4*Alpha,10+2*Graph);
15155                         PRINT USING "ZZZ";Pen
15160                         Pen=Pen+1
15165                     NEXT Graph
15170                 NEXT Alpha
15175                 ALPHA PEN Black
15180                 KEY LABELS PEN Blue
15185                 PRINT PEN Black
15190             SUBEND
15195 Dump:       SUB Dump
15200                 !  Description:
15205                 !      This subprogram dumps the graphics contents of the CRT to the printer.  This facilitates
15210                 !      the printing of the histogram and profile plots.  The CSUB binary subprogram is used to
15215                 !      transfer the colorized plots to the color paint jet printer.
15220                 OUTPUT PRT USING "#,@"
15225                 IF NOT (INMEM("Gdump_colored")) THEN LOADSUB ALL FROM "CDUMP6"
15230                 IF NOT (INMEM("Bstore")) THEN LOADSUB ALL FROM "BPLOT6"
15235                 IF NOT (INMEM("Bload")) THEN LOADSUB ALL FROM "BPLOT6"
15240                !OUTPUT PRT USING "#,5/"
```

```
15245                  !CALL Gdump_colored(CRT,PRT,"NORMAL",180,"OFF","DITHER")
15250                  !CALL Gdump_colored(CRT,PRT,"ROTATE",90,"ON","ERRDIF")
15255                   CALL Gdump_colored(CRT,PRT,"NORMAL",180,"ON","DITHER")
15260              SUBEND
15265 Read_symbols:  SUB Read_symbols(Symbols(*))
15270                  !  Description:
15275                  !      This subprogram defines 5 geometric symbols to be used with the SYMBOL statement.  The
15280                  !      symbols provided are as follows:  Square,Octagon,Diamond, and Triangles (upwards & downwards
15285                  !      pointing triangles).  All of the symbols have a dot added to their center.
15290                  !  Variables:
15295                  !      Symbols(*)  Array of Symbol arrays.  Each symbol arrays contains a distinct geometric symbol.
15300                  !      Symbol(*)   Array of coordinates which when connected produce a distinct geometric symbol.
15305                  !      Dot(*)      Array of coordinates which produce a dot.  The dot symbol is added to all symbols.
15310                  !      Noc         The number of coordinates in a symbol.
15315                  !      S           Used to index the Symbols array.
15320                  OPTION BASE 1
15325                  REAL Symbol(20,3),Dot(2,3)
15330                  READ Dot(*)
15335                  FOR S=1 TO 5
15340                      READ Noc
15345                      REDIM Symbol(Noc,3)
15350                      READ Symbol(*)
15355                      MAT Symbols(S,1:Noc,*)= Symbol
15360                      MAT Symbols(S,Noc+1:Noc+2,*)= Dot
15365                      Symbols(S,0,1)=Noc+2
15370                  NEXT S
15375 Dot:             DATA     4.5, 7.5,-2,  4.5, 7.5,-1
15380 Square:          DATA 5,  0.5, 3.5,-2,  8.5, 3.5,-1,  8.5,11.5,-1,  0.5,11.5,-1,  0.5,3.5,-1
15385 Octagon:         DATA 9,  0.5, 5.5,-2,  2.5, 3.5,-1,  6.5, 3.5,-1,  8.5, 5.5,-1,  8.5,9.5,-1,  6.5,11.5,-1,
                             2.5,11.5,-1,  0.5,9.5,-1,  0.5,5.5,-1
15390 Diamond:         DATA 5, -0.5, 7.5,-2,  4.5, 2.5,-1,  9.5, 7.5,-1,  4.5,12.5,-1, -0.5,7.5,-1
15395 Utriangle:       DATA 4,  0.5, 4.5,-2,  8.5, 4.5,-1,  4.5,13.5,-1,  0.5, 4.5,-1
15400 Dtriangle:       DATA 4,  0.5,10.5,-2,  8.5,10.5,-1,  4.5, 1.5,-1,  0.5,10.5,-1
15405              SUBEND
15410 Graph:       !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
15415 Setup_graph:   SUB Setup_graph(Array(*),Image$(*),Paxis,Symbols(*))
15420                  !  Description:
15425                  !      This subprogram sets up nine empty plots on the CRT screen.  Four plots are profile plots
15430                  !      while the other five plots are histogram plots.  The profile and histogram plots provided are
15435                  !      as follows:      Graph#       Type                 Description
15440                  !                         1        Histogram #1      U frequency data in MHz.
15445                  !                         2        Histogram #2      V frequency data in MHz.
15450                  !                         3        Histogram #3      W frequency data in MHz.
15455                  !                         4        Histogram #4      Analog Channel #1 data in volts.
15460                  !                         5        Histogram #5      Analog Channel #2 data in volts.
15465                  !                         6        Profile Plot #1   Velocity Averages & SDVs vs. Traverse Position.
15470                  !                         7        Profile Plot #2   Voltage  Averages & SDVs vs. Traverse Position.
15475                  !                         8        Profile Plot #3   Temperature Average & SDV vs. Traverse Position.
15480                  !                         9        Profile Plot #4   Velocity Shear Stress Terms vs. Traverse Position.
15485                  !  Variables:
15490                  !      Array(*)     Array containing the plot positions and scales.
15495                  !      Image$(*)    String array containing image formats for the axes labeling.
15500                  !      Wndw(*)      Array containing the plot's scales.
15505                  !      Vwprt(*)     Array containing the plot's CRT position.
15510                  !      Xdiv(*)      Array containing the number of X divisions for the plot's X axis.
15515                  !      Ydiv(*)      Array containing the number of Y divisions for the plot's Y axis.
15520                  !      Xlabel$(*)   String array containing labels for the X axis.
15525                  !      Ylabel$(*)   String array containing labels for the Y axis.
15530                  !      Title$(*)    String array containing labels for the Plots.
15535                  !      Ximage$(*)   String array containing image formats for the X axis labeling.
15540                  !      Yimage$(*)   String array containing image formats for the Y axis labeling.
15545                  !      Legend$(*)   String array containing labels for each symbol in a profile plot.
15550                  !      Symbols(*)   Array of Symbol arrays.  Each symbol arrays contains a distinct geometric symbol.
15555                  !      G            Used as an index to the above arrays.  Specifies one of nine plots.
15560                  !      I            Used an an index to the Legend$ array.
15565                  OPTION BASE 1
15570                  COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
15575                  COM /Graph2/ Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
15580                  COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
15585                  COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
15590                  MAT Wndw= Array(61:69,*)
15595                  MAT Vwprt= Array(71:79,*)
15600                  MAT Xdiv(1:5)= Array(81:85,1)
15605                  MAT Xdiv(6:9)= Array(81:84,3)
15610                  MAT Ydiv(1:5)= Array(81:85,2)
15615                  MAT Ydiv(6:9)= Array(81:84,4)
15620                  MAT Ximage$= Image$(61:69,1)
15625                  MAT Yimage$= Image$(61:69,3)
15630                  FOR G=1 TO 9
15635                      READ G,Xlabel$(G)
```

```
15640                      FOR I=1 TO SIZE(Legend$,2)
15645                          READ Legend$(G,I)
15650                      NEXT I
15655                      SELECT G
15660                      CASE 1 TO 5
15665                          Ylabel$(G)=""
15670                      CASE 6 TO 9
15675                          Ylabel$(G)=CHR$(NUM("X")+Paxis-1)
15680                      END SELECT
15685                      CALL Set_up(G,Symbols(*))
15690                  NEXT G
15695                  SUBEXIT
15700                  !    G, X axis Label                          ,       Symbol #1...5 labels
15705                  DATA 1, ""                                    ,     "",     "",      "","",""
15710                  DATA 2, ""                                    ,     "",     "",      "","",""
15715                  DATA 3, ""                                    ,     "",     "",      "","",""
15720                  DATA 4, ""                                    ,     "",     "",      "","",""
15725                  DATA 5, ""                                    ,     "",     "",      "","",""
15730                  DATA 6, "U,V,U',V'   /Uinf"                   ,   "U",    "V",    "U'","V'  /Uinf",""
15735                  DATA 7, "A,B,A',B'   volts"                   ,   "A",    "B",    "A'","B'  volts",""
15740                  DATA 8, "Tt:dR   Uinf:m/s   Uedge:m/s" ,  "Tt:dR","Uinf","Uedge","",""
15745                  DATA 9, "Shear Stress Terms / Uinf^2"  ,"U'V'","V'W'","W'U'  /Uinf^2","",""
15750              SUBEND
15755 Set_up:      SUB Set_up(G,Symbols(*))
15760                  !  Description:
15765                  !      This subprogram clears and then redraws one of nine empty plots on the CRT screen.
15770                  !  Variables:
15775                  !      Wndw(*)       Array containing the plot's scales.
15780                  !      Vwprt(*)      Array containing the plot's CRT position.
15785                  !      Xdiv(*)       Array containing the number of X divisions for the plot's X axis.
15790                  !      Ydiv(*)       Array containing the number of Y divisions for the plot's Y axis.
15795                  !      Xlabel$(*)    String array containing labels for the X axis.
15800                  !      Ylabel$(*)    String array containing labels for the Y axis.
15805                  !      Title$(*)     String array containing labels for the Plots.
15810                  !      Ximage$(*)    String array containing image formats for the X axis labeling.
15815                  !      Yimage$(*)    String array containing image formats for the Y axis labeling.
15820                  !      Legend$(*)    String array containing labels for each symbol in a profile plot.
15825                  !      Symbols(*)    Array of Symbol arrays.  Each symbol arrays contains a distinct geometric symbol.
15830                  !      G             Used as an index to the above arrays.  Specifies one of nine plots.
15835                  OPTION BASE 1
15840                  COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
15845                  COM /Graph2/ Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
15850                  COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
15855                  COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
15860                  DIM L$[80]
15865                  ON ERROR CALL Error
15870                  CSIZE 100*15/1023       ! Select a character labeling size of 15 pixels high.
15875                  ! Define the values for the left,right,bottom,top ends of the horizontal and vertical scales.
15880                  Xmin=Wndw(G,1)
15885                  Xmax=Wndw(G,2)
15890                  Ymin=Wndw(G,3)
15895                  Ymax=Wndw(G,4)
15900                  ! Define the values for the left,right,bottom,top pixel locations for the plot.
15905                  Xpix1=Vwprt(G,1)
15910                  Xpix2=Vwprt(G,2)
15915                  Ypix1=Vwprt(G,3)
15920                  Ypix2=Vwprt(G,4)
15925                  ! Define the step size between grid lines, axis tick marks, and axis labels.
15930                  Xstep=(Xmax-Xmin)/Xdiv(G)
15935                  Ystep=(Ymax-Ymin)/Ydiv(G)
15940                  ! Define the amount of scale X and Y which equals the size of one pixel (picture element).
15945                  Xpixel=(Xmax-Xmin)/(Xpix2-Xpix1)
15950                  Ypixel=(Ymax-Ymin)/(Ypix2-Ypix1)
15955                  ! Clear the plots back ground & plot area and also draw the plots borders, grids, and axes.
15960                  AREA PEN 16*White
15965                 !GOSUB Clear_screen
15970                  AREA PEN 16*White
15975                  GOSUB Back_ground
15980                  AREA PEN 16*White
15985                  GOSUB Plot_area
15990                  GOSUB Scale
15995                  PEN 16*Blue
16000                  GOSUB Axes
16005                  GOSUB Grid
16010                  GOSUB Scale
16015                  PEN 16*Black
16020                  CLIP OFF
16025                  ! Draw the X and Y axis labels.
16030                  GOSUB Ylabel
16035                  GOSUB Xlabel
```

```
16040                   ! Create a legend to define which symbol is used with which data.
16045                   CALL Legend(G,Symbols(*))
16050                   OFF ERROR
16055                   SUBEXIT
16060 Clear_screen:     ! This subroutine fills the entire CRT screen with the specified color.
16065                   VIEWPORT 0/10.23,1279/10.23,0/10.23,1023/10.23
16070                   WINDOW -1.E+9,1.E+9,-1.E+9,1.E+9
16075                   MOVE 0,0
16080                   WINDOW 0,1279,0,1023
16085                   MOVE 0,0
16090                   RECTANGLE 1279,1023,FILL
16095                   RETURN
16100 Back_ground:      ! This subroutine clears the plot's background.
16105                   VIEWPORT (Xpix1-80)/10.23,(Xpix2+15)/10.23,(Ypix1-33)/10.23,(Ypix2+10)/10.23
16110                   WINDOW -1.E+9,1.E+9,-1.E+9,1.E+9
16115                   MOVE 0,0
16120                   WINDOW Xmin,Xmax,Ymin,Ymax
16125                   MOVE Xmin,Ymin
16130                   RECTANGLE (Xmax-Xmin),(Ymax-Ymin),FILL
16135                   RETURN
16140 Plot_area:        ! This subroutine selects part of the CRT plot area and give it scales for the X and Y axes.
16145                   VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
16150                   WINDOW -1.E+9,1.E+9,-1.E+9,1.E+9
16155                   MOVE 0,0
16160                   WINDOW Xmin,Xmax,Ymin,Ymax
16165                   MOVE Xmin,Ymin
16170                   RECTANGLE (Xmax-Xmin),(Ymax-Ymin),FILL
16175                   RETURN
16180 Axes:             ! This subroutine draws the plot's X and Y axes.
16185                   VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-6)/10.23,(Ypix1-1)/10.23
16190                   WINDOW Xmin,Xmax,1,0
16195                   AXES Xstep,2,Xmin,0,1,1,1
16200                   VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix2+1)/10.23,(Ypix2+6)/10.23
16205                   WINDOW Xmin,Xmax,0,1
16210                   AXES Xstep,2,Xmin,0,1,1,1
16215                   VIEWPORT (Xpix1-6)/10.23,(Xpix1-1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16220                   WINDOW 1,0,Ymin,Ymax
16225                   AXES 2,Ystep,0,Ymin,1,1,1
16230                   VIEWPORT (Xpix2+1)/10.23,(Xpix2+6)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16235                   WINDOW 0,1,Ymin,Ymax
16240                   AXES 2,Ystep,0,Ymin,1,1,1
16245                   RETURN
16250 Grid:             ! This subroutine draws the plot's X and Y grid lines.
16255                   VIEWPORT (Xpix1-1)/10.23,(Xpix2+1)/10.23,(Ypix1-1)/10.23,(Ypix2+1)/10.23
16260                   WINDOW Xmin,Xmax,Ymin,Ymax
16265                   LINE TYPE 4
16270                   GRID Xstep,Ystep,Xmin,Ymin
16275                   LINE TYPE 1
16280                   RETURN
16285 Scale:            VIEWPORT Xpix1/10.23,Xpix2/10.23,Ypix1/10.23,Ypix2/10.23
16290                   WINDOW Xmin,Xmax,Ymin,Ymax
16295                   RETURN
16300 Xlabel:           ! This subroutine labels the X axis and also names the X axis.
16305                   LORG 5
16310                   FOR X=Xmin TO Xmax+Xstep/100 STEP Xstep
16315                       MOVE X,Ymin-14*Ypixel
16320                       OUTPUT L$ USING Ximage$(G);X
16325                       LABEL TRIM$(L$)
16330                   NEXT X
16335                   MOVE (Xmin+Xmax)/2,Ymin-27*Ypixel
16340                  !LABEL Xlabel$(G)
16345                   RETURN
16350                   IF G=8 THEN
16355                       LORG 5
16360                       DEG
16365                       LDIR 45
16370                       MOVE (Xmin+Xmax)/2,(Ymax+Ymin)/2
16375                       CSIZE 100*100/1023
16380                       LABEL "VOID"
16385                       CSIZE 100*15/1023
16390                       LDIR 0
16395                   END IF
16400                   RETURN
16405 Ylabel:           ! This subroutine labels the Y axis and also names the Y axis.
16410                   LORG 8
16415                   Len=0
16420                   FOR Y=Ymin TO Ymax+Ystep/100 STEP Ystep
16425                       MOVE Xmin-7*Xpixel,Y
16430                       OUTPUT L$ USING Yimage$(G);Y
16435                       LABEL TRIM$(L$)
```

```
16440                   Len=MAX(Len,LEN(TRIM$(L$)))
16445               NEXT Y
16450               LORG 5
16455               MOVE Xmin-(18+7*Len)*Xpixel,(Ymin+Ymax)/2                              !+20*Ypixel
16460               LABEL Ylabel$(G)
16465               RETURN
16470           SUBEND
16475 Legend:   SUB Legend(G,Symbols(*))
16480               !  Description:
16485               !      This subprogram produces a legend within one of the nine plots on the CRT screen.
16490               !  Variables:
16495               !      Wndw(*)       Array containing the plot's scales.
16500               !      Vwprt(*)      Array containing the plot's CRT position.
16505               !      Xdiv(*)       Array containing the number of X divisions for the plot's X axis.
16510               !      Ydiv(*)       Array containing the number of Y divisions for the plot's Y axis.
16515               !      Xlabel$(*)    String array containing labels for the X axis.
16520               !      Ylabel$(*)    String array containing labels for the Y axis.
16525               !      Title$(*)     String array containing labels for the Plots.
16530               !      Ximage$(*)    String array containing image formats for the X axis labeling.
16535               !      Yimage$(*)    String array containing image formats for the Y axis labeling.
16540               !      Legend$(*)    String array containing labels for each symbol in a profile plot.
16545               !      Symbols(*)    Array of Symbol arrays.  Each symbol arrays contains a distinct geometric symbol.
16550               !      Symbol(*)     Array of coordinates which when connected produce a distinct geometric symbol.
16555               !      G             Used as an index to the above arrays.  Specifies one of nine plots.
16560               !      S             Used to index the Legend$ array.
16565               !      Noc           The number of coordinates in a symbol.
16570               !      Len           Total Length of all Legend$ array elements.
16575               OPTION BASE 1
16580               COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
16585               COM /Graph2/ Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
16590               COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
16595               COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
16600               DIM Symbol(20,3)
16605               VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
16610               WINDOW Vwprt(G,1),Vwprt(G,2),Vwprt(G,3),Vwprt(G,4)
16615               CLIP OFF
16620               CSIZE 100*15/1023      ! Select a character labeling size of 15 pixels high.
16625               LORG 2
16630               ! Calculate the total length of all of the symbol labels.
16635               Len=0
16640               FOR S=1 TO SIZE(Legend$,2)
16645                   IF LEN(Legend$(G,S)) THEN
16650                       Len=Len+LEN(TRIM$(Legend$(G,S)))+2.2
16655                   END IF
16660               NEXT S
16665               X=(Vwprt(G,1)+Vwprt(G,2))/2
16670               Y=(Vwprt(G,3)+Vwprt(G,4))/2
16675               MOVE X,Y
16680               X=(Vwprt(G,1)+Vwprt(G,2))/2-5*Len+10
16685               Y=Vwprt(G,3)-28
16690               ! For each symbol put up a sample symbol and its label.
16695               FOR S=1 TO SIZE(Legend$,2)
16700                   IF LEN(Legend$(G,S))=0 THEN 16825
16705                   Noc=Symbols(S,0,1)
16710                   REDIM Symbol(Noc,3)
16715                   MAT Symbol= Symbols(S,1:Noc,*)
16720                   ! Define the colors for symbol filling and edge drawing.
16725                   SELECT S
16730                   CASE 1
16735                       AREA PEN 16*Red
16740                       PEN 16*Black
16745                   CASE 2
16750                       AREA PEN 16*Yellow
16755                       PEN 16*Black
16760                   CASE 3
16765                       AREA PEN 16*Green
16770                       PEN 16*Black
16775                   CASE 4
16780                       AREA PEN 16*Blue
16785                       PEN 16*Black
16790                   END SELECT
16795                   MOVE X,Y                         ! Move to the place of next symbol.
16800                   SYMBOL Symbol(*),FILL,EDGE       ! Draw the next symbol.
16805                   X=X+12                           ! Move the X placement to the right 12 pixels.
16810                   MOVE X,Y-1                       ! Move to the place of next label.
16815                   LABEL Legend$(G,S)               ! Draw the next label.
16820                   X=X+10*LEN(Legend$(G,S))+10      ! Move the X placement to the right 10+10*Len pixels
16825               NEXT S
16830           SUBEND
16835 Histo:    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
16840 Rt_histo:     SUB Rt_histo(@Lvdas,Symbols(*),Repeat,Kbd$)
16845                 ! Description:
16850                 !     This subprogram plots real time histograms within five of the nine plots on the CRT screen.
16855                 !     The histogram data are acquired from the LVDAS over a specified acquisition time.
16860                 ! Variables Defined in Main Program:
16865                 !    Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
16870                 ! Local Variables:
16875                 !    Histo(*)  Array of bin numbers, old histogram bin heights, and new histogram bin heights.
16880                 !    Nbins     Number of bins in the Histo(*).
16885                 !    Bin       2^Bin is the bin width of individual histogram vertical bars.
16890                 !    Min       Minimum value for histogram.  Left  side of histogram scale.
16895                 !    Max       Maximum value for histogram.  right side of histogram scale.
16900                 !    F1        Upper 16bits of integerized Min.
16905                 !    F2        Lower 16bits of integerized Min.
16910                 !    A1        Upper 16bits of integerized histogram acquisition time.
16915                 !    A2        Lower 16bits of integerized histogram acquisition time.
16920                 !    Nnew      Number of samples in the most up to date histogram.
16925                 !    Nold      Number of samples in the previous histogram.
16930                 !    N(*)      Number of samples for each histogram of the five separate channels.
16935                 !    Channel   Used to select the LVDAS channel that will be sampled for a histogram.
16940                 !    Kw        Converts Hz to MHz or raw data to volts.
16945                 !    Ww        Window width of each vertical histogram bar.
16950                 !    Old       Histogram height of previous histogram at a particular bin.
16955                 !    New       Histogram height of current  histogram at a particular bin.
16960                 !    X1        Horizontal position of histogram rectangle.
16965                 !    Y1        Vertical  position of histogram rectangle.
16970                 !    X2        Horizontal width of histogram rectangle.
16975                 !    Y2        Vertical  width of histogram rectangle.
16980                 !    I         Used as an index to the Histo(*).  Specifies one of Nbins bins.
16985                 !    G         Used as an index to the graphics arrays.  Specifies one of nine plots.
16990                 OPTION BASE 1
16995                 COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
17000                 COM /Graph2/ Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
17005                 COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
17010                 COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
17015                 INTEGER Histo(1000,3),Nplots,Nbins,F1,F2,A1,A2
17020                 REAL Nnew,Nold,N(5)
17025                 ! Clear all of the histogram data within the LVDAS.
17030                 OUTPUT @Lvdas USING "AA";"CA"
17035                 ! Draw new plots for the five histograms.
17040                 FOR Channel=1 TO 5
17045                     CALL Set_up(Channel,Symbols(*))
17050                 NEXT Channel
17055                 ! Calculate the acquisition time.  0.1*10000000 will give an acquisition of 0.1 seconds.
17060                 CALL Convert2words(.1*10000000,A1,A2)                          ! Atime=.1 seconds
17065                 ! Enable the keyboard to terminate histogram plotting.
17070                 ON KBD GOSUB Hdone
17075                 REPEAT
17080                     FOR Channel=1 TO 5
17085                         G=Channel
17090                         SELECT Channel
17095                         CASE 1,2                                    ! Channels 1,2,3 are for LDV frequency data.
17100                             Kw=1000000                              ! Converts Hz to MHz.
17105                             Min=Kw*Wndw(G,1)                        ! Minimum frequency for left  histogram scale.
17110                             Max=Kw*Wndw(G,2)                        ! Maximum frequency for right histogram scale.
17115                             Bin=INT(LGT((Max-Min)/100)/LGT(2))+1    ! 2^Bin is the window width of each vertical bar.
17120                             Ww=2^Bin                                ! Window width of each vertical histogram bar.
17125                             CALL Convert2words(Min,F1,F2)
17130                         CASE 4                                      ! Channels 4,5 are for analog voltage data.
17135                             Kw=32768/5                              ! Converts raw data to volts.
17140                             Min=Kw*Wndw(G,1)                        ! Minimum voltage for left  histogram scale.
17145                             Max=Kw*Wndw(G,2)                        ! Maximum voltage for right histogram scale.
17150                             Bin=INT(LGT((Max-Min)/100)/LGT(2))+1    ! 2^Bin is the window width of each vertical bar.
17155                             Ww=2^Bin                                ! Window width of each vertical histogram bar.
17160                             CALL Convert2words(Min,F1,F2)
17165                         CASE ELSE
17170                             GOTO 17350
17175                         END SELECT
17180 Hsend:                  ! Tell the LVDAS to Take a Histogram.
17185                         OUTPUT @Lvdas USING "AA,6(W)";"TH",F1,F2,Bin,A1,A2,Channel
17190 Henter:                 ! Enter number of bins in the histogram.
17195                         ENTER @Lvdas USING "#,W";Nbins
17200                         ! Redimension the Histo(*) and the enter the histogram data.
17205                         IF Nbins>0 THEN
17210                             REDIM Histo(Nbins,3)
17215                             ENTER @Lvdas USING "#,W";Histo(*)
17220                         END IF
17225                         ! Enter the number of samples for the previous and current histogram.
17230                         ENTER @Lvdas USING "#,W";Nnew,Nold
17235 Hplot:                  ! Scale part of the CRT for the histogram plotting.
```

```
17240                   VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
17245                   WINDOW Kw*Wndw(G,1),Kw*Wndw(G,2),Wndw(G,3),Wndw(G,4)
17250                   Xpixel=Kw*(Wndw(Channel,2)-Wndw(Channel,1))/(Vwprt(Channel,2)-Vwprt(Channel,1))
17255                   N1=N(Channel)
17260                   N2=N(Channel)-Nold+Nnew
17265                   N(Channel)=N(Channel)-Nold+Nnew
17270                   PEN 16*Aqua          ! Select the pen for the histogram bars edge.
17275                   AREA PEN 16*Aqua     ! Select the pen for the histogram bars fill.
17280                   FOR I=1 TO Nbins
17285                     Old=MIN(Histo(I,3),Wndw(Channel,4))
17290                     New=MIN(Histo(I,2),Wndw(Channel,4))
17295                     X1=Histo(I,1)*Ww+Min            ! Calculate histogram bar horizontal position.
17300                     X2=Ww                           ! Calculate histogram bar horizontal width.
17305                     Y1=Old                          ! Calculate histogram bar vertical position.
17310                     Y2=New-Old                      ! Calculate histogram bar vertical width.
17315                     IF X1<Kw*Wndw(G,1) THEN X1=Kw*Wndw(G,1)         ! If X1<Xmin then set X1=Xmin
17320                     IF X1>Kw*Wndw(G,2)-X2 THEN X1=Kw*Wndw(G,2)-X2   ! If X1>Xmax then set X1=Xmax
17325                     MOVE X1,Y1
17330                     CONTROL CRT,14;6                  ! Change to complimentary drawing mode.
17335                     RECTANGLE X2-Xpixel,Y2,FILL,EDGE ! Draw the rectangle representing one bar of the bargraph.
17340                     CONTROL CRT,14;3                  ! Switch back to dominant drawing mode.
17345                   NEXT I
17350                 NEXT Channel
17355                 Kbd$=KBD$
17360               UNTIL Kbd$<>"" OR NOT Repeat              ! Quit if any key on the keyboard has been pressed.
17365               SUBEXIT
17370 Hdone:        Done=1
17375               RETURN
17380             SUBEND
17385 Pt_histo:     SUB Pt_histo(Symbols(*),Run,File,Pos,INTEGER Nsam)
17390               !  Description:
17395               !     This subprogram plots post time histograms within five of the nine plots on the CRT screen.
17400               !     The histogram data are acquired from the LVDAS over a specified acquisition time.
17405               !  Variables Defined in Main Program:
17410               !     Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*),Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
17415               !     Ui(*),Vi(*),Wi(*),Ai(*),Bi(*)
17420               !  Local Variables:
17425               !     Histo(*)  Array of histogram bin heights indexed by bin number.
17430               !     Data(*)   Array of instantaneous U,V,W velocity or A,B voltage data.
17435               !     Nsam      Number of samples acquired.
17440               !     Xmin      Minimum value for histogram.  Left  side of histogram scale.
17445               !     Xmax      Maximum value for histogram.  right side of histogram scale.
17450               !     Xwin      Window width of each vertical histogram bar.
17455               !     K         Used as an index to the above arrays.
17460               !     L         Used as an index to the Histo(*).  Specifies one of 100 bins.
17465               !     Xpixel    Horizontal length of one picture on the CRT in scale units.
17470               !     Channel   Selects one of the 5 channels of Ui(*),Vi(*),Wi(*),Ai(*),Bi(*) data.
17475               !     G         Used as an index to the graphics arrays.  Specifies one of nine plots.
17480               OPTION BASE 1
17485               COM /Data2/ REAL Ui(1000),Vi(1000),Wi(1000),Ai(1000),Bi(1000),Ii(1000),Ci(1000)
17490               COM /Graph1/ Wndw(*),Vwprt(*),Xdiv(*),Ydiv(*),Xlabel$(*),Ylabel$(*)
17495               COM /Graph2/ Title$(*),Ximage$(*),Yimage$(*),Legend$(*)
17500               COM /Color1/ Clear,Black,Red,Yellow,Green,Cyan,Blue,Magenta
17505               COM /Color2/ White,Olive,Aqua,Royal,Maroon,Brick,Brown,Gray
17510               INTEGER Histo(0:100)
17515               REAL Data(1000)
17520               REDIM Data(Nsam)
17525               FOR Channel=1 TO 5
17530                 ! Fill the data array with Ui(*),Vi(*),Wi(*),Ai(*), or Bi(*) depending on Channel.
17535                 G=Channel
17540                 IF Channel=1 THEN MAT Data= Ui
17545                 IF Channel=2 THEN MAT Data= Vi
17550                 IF Channel=3 THEN MAT Data= Wi
17555                 IF Channel=4 THEN MAT Data= Ai
17560                 IF Channel=5 THEN MAT Data= Bi
17565                 ! Draw a new empty histogram plot.
17570                 CALL Set_up(Channel,Symbols(*))
17575 Hsort:          Xmin=Wndw(Channel,1)
17580                 Xmax=Wndw(Channel,2)
17585                 Xwin=(Xmax-Xmin)/100
17590                 ! Sort the data into a histogram.
17595                 MAT Data= Data-(Xmin)
17600                 MAT Data= Data/((Xmax-Xmin)/100)
17605                 MAT Histo= (0)
17610                 FOR K=1 TO Nsam
17615                   L=MAX(MIN(Data(K),100),0)
17620                   Histo(L)=Histo(L)+1
17625                 NEXT K
17630 Hplot:          ! Scale part of the CRT for histogram plotting.
17635                 VIEWPORT Vwprt(G,1)/10.23,Vwprt(G,2)/10.23,Vwprt(G,3)/10.23,Vwprt(G,4)/10.23
```

```
17640                 WINDOW 0,100,Wndw(G,3),Wndw(G,4)
17645                 Xpixel=(100-0)/(Vwprt(Channel,2)-Vwprt(Channel,1))
17650                 ! Draw the histogram.
17655                 FOR K=0 TO 100
17660                     IF Histo(K) THEN
17665                         MOVE K-.5,0
17670                         AREA PEN 16*Green
17675                         RECTANGLE 1-Xpixel,Histo(K),FILL
17680                     END IF
17685                 NEXT K
17690             NEXT Channel
17695             SUBEXIT
17700         SUBEND
17705         CSUB Gdump_colored(From_ds,To_ds,OPTIONAL Rotate$,INTEGER Resolution,Background$,Algorithm$)
17710         CSUB Bload(INTEGER A(*),Xpixels,Ypixels,OPTIONAL INTEGER Rule,REAL Xstart,Ystart)
17715         CSUB Bstore(INTEGER A(*),Xpixels,Ypixels,OPTIONAL INTEGER Rule,REAL Xstart,Ystart)
DONE
```

APPENDIX C

DATA REDUCTION AND COORDINATE SYSTEM TRANSFORMATION EQUATIONS.

APPENDIX C

Data Reduction and Coordinate System Transformation Equations.

SECTION **TABLE OF CONTENTS** Page

FIGURE **LIST OF FIGURES** Page

1. Introduction

The purpose of this write-up is to describe the data reduction performed on the raw data acquired from the Laser Velocimeter Data Acquisition System. The digital Macrodyne data are converted from raw 16bit integer words into frequencies. The frequency results are in turn converted into particle velocities. The analog data are converted from raw two's complement 16bit integer words into voltages. Example types for the analog data might originate from such sources as temperature probes, laser fluorescence anemometers, hot wire anemometers, etc.

Section 2 contains a list of variables that are used throughout this write-up. A brief description of each variable is provided along with the corresponding variable name that is used in the software program. (NOTE: This chapter has been written for three component LDV systems. The delivered system is a two component system. Therefore, the third component W of the three components U,V,W is not measured.)

Velocities are measured in "Laser Coordinates" directly. That is, the measured velocity of each component is parallel to a vector which is orthogonal to the fringe planes in the probe volume. These vectors may or may not be parallel to the tunnel coordinate system. If they are not, then it is desirable to convert the velocities from "Laser Coordinates" to "Tunnel Coordinates." In other words, a coordinate system transformation needs to be applied to the measured velocities to obtain velocities in tunnel coordinates. Section 3 describes how this laser to tunnel coordinate system transformation is performed. (NOTE: The delivered system is a two component system whose laser beam pairs have been orientated orthogonally to the wind tunnel's X,Y,Z axes. Therefore, velocities measured in "Laser Coordinates" will be equal to velocities transformed to "Tunnel Coordinates" . For this reason, the the "Laser" to "Tunnel" coordinate system transformations have not been included in the "3.5'HWT" data acquisition programs listed in Appendixes A & B.)

In some cases it is preferred to perform an additional coordinate system transformation

to obtain velocities in "Model Coordinates." For example, if the model is at an angle of attack, then the model's coordinate system would be at rotation with respect to the tunnel's coordinate system. Other model attitude angles in addition to the angle of attack, such as roll and yaw, can be used determine the transformation required to convert from tunnel to model coordinates. Section 4 describes how this tunnel to model coordinate system transformation is performed.

Section 5 contains the equations that are used to calculate the average, standard deviation, as well as normal and shear stress terms for the velocity and voltage data. Equations are included for both the original and transformed to coordinate systems. Normal text is used in the equations for variables that represent the original coordinate system while italicized text is used for variables that represent the transformed to coordinate system.

Section 6 contains the equations that are used to convert average, standard deviation, as well as normal and shear stress terms from the original to the transformed to coordinate system.

Section 7 contains proofs demonstrating that we can perform the coordinate system transformations on the reduced averaged data without having to perform the transformation on the instantaneous values. This saves costly run time because there are typically thousands of instantaneous values that contribute one averaged value.

Section 8 shows how the equations of section 6 can be represented in matrix notation. The matrix notation for the coordinate system transformation is an elegant way to show the multitude of complex equations in compact and concise format.

2. List of Variables

The following is a list of variables that are used throughout this write-up. A brief description of each variable is provided along with the corresponding variable name that is used in the software program. Normal text style (not italicized) indicates the original coordinate system while italicized text style indicates a transformed to coordinate system.

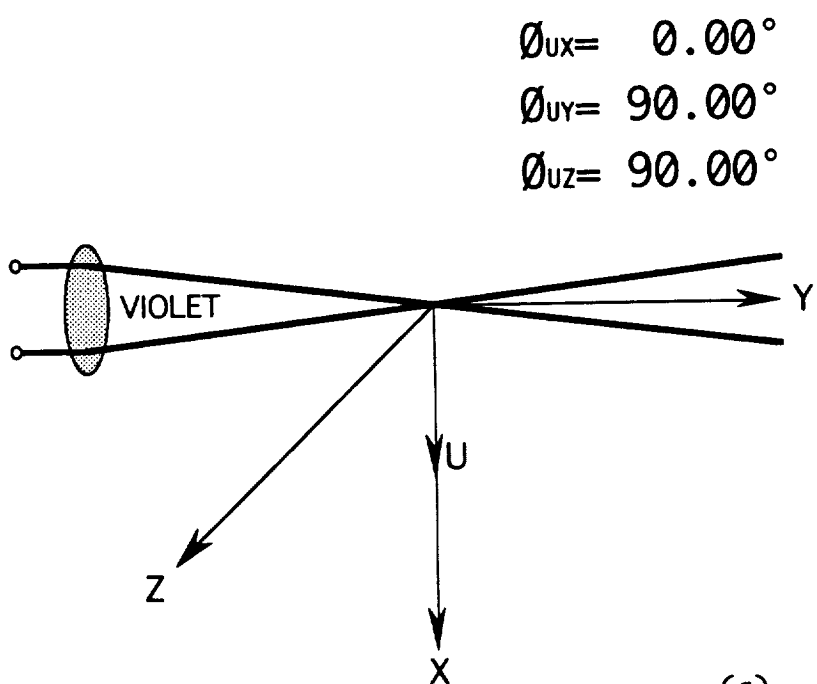
Original	Transformed	Description	Variable
$\mathrm{U_i}$	U_i	Instantaneous U velocity.	`Ui(I)`
$\mathrm{V_i}$	V_i	Instantaneous V velocity.	`Vi(I)`
$\mathrm{W_i}$	W_i	Instantaneous W velocity.	`Wi(I)`
$\mathrm{A_i}$	A_i	Instantaneous A voltage.	`Ai(I)`
$\mathrm{B_i}$	B_i	Instantaneous B voltage.	`Bi(I)`
$\overline{\mathrm{U}}$	$\overline{U}$	Average U velocity.	`U`
$\overline{\mathrm{V}}$	$\overline{V}$	Average V velocity.	`V`
$\overline{\mathrm{W}}$	$\overline{W}$	Average W velocity.	`W`
$\overline{\mathrm{A}}$	$\overline{A}$	Average A voltage.	`A`
$\overline{\mathrm{B}}$	$\overline{B}$	Average B voltage.	`B`
$\mathrm{U'}$	U'	U velocity standard deviation.	`U1`
$\mathrm{V'}$	V'	V velocity standard deviation.	`V1`
$\mathrm{W'}$	W'	W velocity standard deviation.	`W1`
$\mathrm{A'}$	A'	A voltage standard deviation.	`A1`
$\mathrm{B'}$	B'	B voltage standard deviation.	`B1`
$\overline{\mathrm{A'A'}}$	$\overline{A'A'}$	A-A normal stress term.	`A1a1`
$\overline{\mathrm{B'B'}}$	$\overline{B'B'}$	B-B normal stress term.	`B1b1`
$\overline{\mathrm{A'B'}}$	$\overline{A'B'}$	A-B shear stress term.	`A1b1`
$\overline{\mathrm{U'A'}}$	$\overline{U'A'}$	U-A shear stress term.	`U1a1`
$\overline{\mathrm{V'A'}}$	$\overline{V'A'}$	V-A shear stress term.	`V1a1`
$\overline{\mathrm{W'A'}}$	$\overline{W'A'}$	W-A shear stress term.	`W1a1`

Original	Transformed	Description	Variable
$\overline{U'U'}$	$\overline{U'U'}$	U-U normal stress term.	`U1u1`
$\overline{U'V'}$	$\overline{U'V'}$	U-V shear stress term.	`U1v1`
$\overline{U'W'}$	$\overline{U'W'}$	U-W shear stress term.	`U1w1`
$\overline{V'U'}$	$\overline{V'U'}$	V-U shear stress term.	`V1u1`
$\overline{V'V'}$	$\overline{V'V'}$	V-V normal stress term.	`V1v1`
$\overline{V'W'}$	$\overline{V'W'}$	V-W shear stress term.	`V1w1`
$\overline{W'U'}$	$\overline{W'U'}$	W-U shear stress term.	`W1u1`
$\overline{W'V'}$	$\overline{W'V'}$	W-V shear stress term.	`W1v1`
$\overline{W'W'}$	$\overline{W'W'}$	W-W normal stress term.	`W1w1`
X		Tunnel or Model X axis	`X`
Y		Tunnel or Model Y axis	`Y`
Z		Tunnel or Model Z axis	`Z`
$\varnothing_{UX}$		Angle between Laser U and Tunnel X	`ThetaAU`
$\varnothing_{UY}$		Angle between Laser U and Tunnel Y	`ThetaAV`
$\varnothing_{UZ}$		Angle between Laser U and Tunnel Y	`ThetaAW`
$\varnothing_{VX}$		Angle between Laser V and Tunnel X	`ThetaBU`
$\varnothing_{VY}$		Angle between Laser V and Tunnel Y	`ThetaBV`
$\varnothing_{VZ}$		Angle between Laser V and Tunnel Y	`ThetaBW`
$\varnothing_{WX}$		Angle between Laser W and Tunnel X	`ThetaCU`
$\varnothing_{WY}$		Angle between Laser W and Tunnel Y	`ThetaCV`
$\varnothing_{WZ}$		Angle between Laser W and Tunnel Y	`ThetaCW`
α_1		Model angle of attack	`Alpha(1)`
α_2		Model angle of roll	`Alpha(2)`
α_3		Model angle of yaw	`Alpha(3)`
$\mathbf{J}_{3X3}$		Coordinate system transformation matrix	---
$\mathbf{K}_{3X3}$		Coordinate system transformation matrix	---
$\mathbf{K}_{9X9}$		Coordinate system transformation matrix	---

3. Laser to Tunnel Coordinate System Transformation

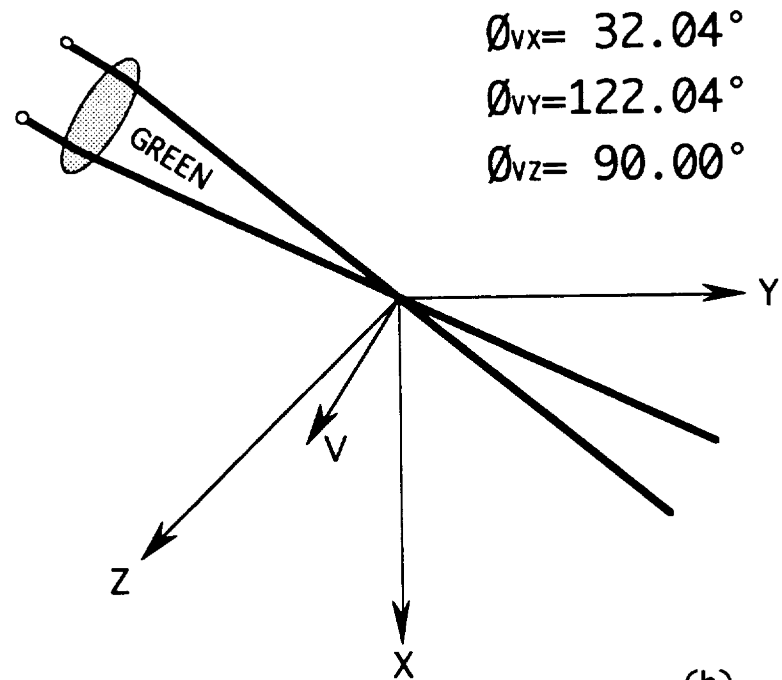
Velocities are measured in "Laser Coordinates" directly. That is, the measured velocity of each component is parallel to a vector which is orthogonal to the fringe planes in the probe volume. These vectors may or may not be parallel to the tunnel coordinate system. If they are not, then it is desirable to convert the velocities from "Laser Coordinates" to "Tunnel Coordinates." In other words, a coordinate system transformation needs to be applied to the measured velocities to obtain velocities in tunnel coordinates. This section describes how this laser to tunnel coordinate system transformation is performed.

Tunnel Coordinate have the axes label as X,Y,Z while velocities measured in laser coordinates typically named U,V,W. The direction of each of the measured velocities in laser coordinates can be defined in terms of the angle it is off of the tunnel coordinate axes. The three angles $\varnothing_{UX}$, $\varnothing_{UY}$, $\varnothing_{UZ}$ define the angular relationship between measured U velocities in laser coordinates and the axes X,Y,Z of the tunnel coordinate system (Fig. 1a).



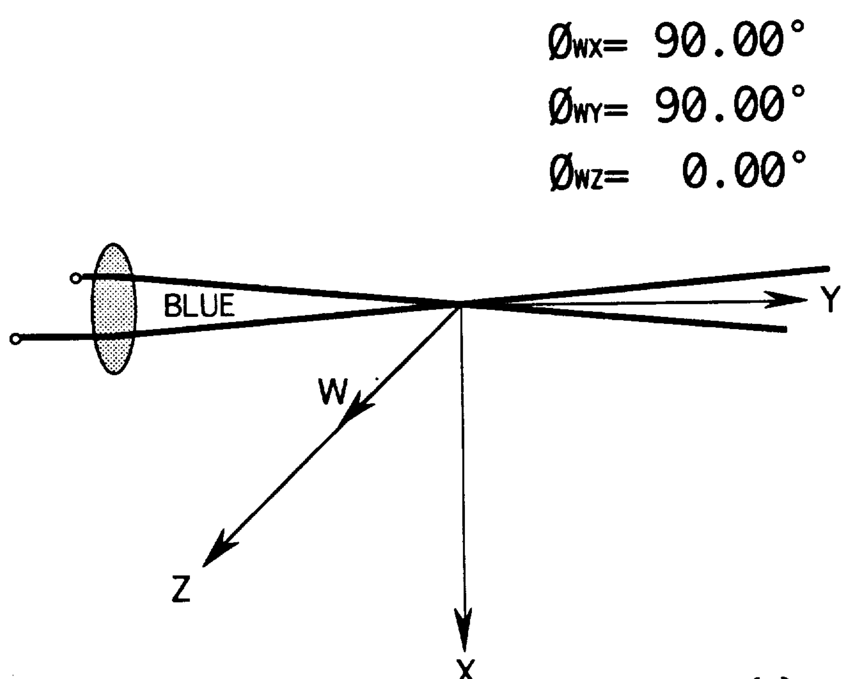
(a)

The three angles $\varnothing_{VX}$, $\varnothing_{VY}$, $\varnothing_{VZ}$ define the angular relationship between measured V velocities in laser coordinates and the axes X,Y,Z of the tunnel coordinate system (Fig. 1b)



(b)

The three angles $\varnothing_{WX}$, $\varnothing_{WY}$, $\varnothing_{WZ}$ define the angular relationship between measured W velocities in laser coordinates and the axes X,Y,Z of the tunnel coordinate system (Fig. 1c)



(c)

When a particle travels through the probe volume, its velocity is measured as U,V,W in

laser coordinates. However, it is desired to have these velocities (U,V,W) transformed to tunnel coordinate velocities *(U,V,W)*. Each of tunnel coordinate velocities *U,V,W* would be parallel to its X,Y,Z tunnel axis. The laser coordinate velocities U,V,W can be defined in terms of the tunnel coordinate velocities using the follow equations:

$$\mathrm{U} = U\cos(\varnothing_{UX}) + V\cos(\varnothing_{UY}) + W\cos(\varnothing_{UZ})$$
$$\mathrm{V} = U\cos(\varnothing_{VX}) + V\cos(\varnothing_{VY}) + W\cos(\varnothing_{VZ})$$
$$\mathrm{W} = U\cos(\varnothing_{WX}) + V\cos(\varnothing_{WY}) + W\cos(\varnothing_{WZ})$$

The coefficients of the of the three equations above can be used to define the coordinate transformation matrix $\mathbf{J}_{3x3}$ as shown below:

$$J_{11}=\cos(\varnothing_{UX}) \quad J_{12}=\cos(\varnothing_{UY}) \quad J_{13}=\cos(\varnothing_{UZ})$$
$$J_{21}=\cos(\varnothing_{VX}) \quad J_{22}=\cos(\varnothing_{VY}) \quad J_{23}=\cos(\varnothing_{VZ})$$
$$J_{31}=\cos(\varnothing_{WX}) \quad J_{32}=\cos(\varnothing_{WY}) \quad J_{33}=\cos(\varnothing_{WZ})$$

$$\mathbf{J}_{3X3} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} = \begin{bmatrix} \cos(\varnothing_{UX}) & \cos(\varnothing_{UY}) & \cos(\varnothing_{UZ}) \\ \cos(\varnothing_{VX}) & \cos(\varnothing_{VY}) & \cos(\varnothing_{VZ}) \\ \cos(\varnothing_{WX}) & \cos(\varnothing_{WY}) & \cos(\varnothing_{WZ}) \end{bmatrix}$$

The coordinate transformation matrix $\mathbf{J}_{3x3}$ can be used to convert tunnel coordinate velocities to laser coordinate velocities.

$$\begin{bmatrix} \mathrm{U} \\ \mathrm{V} \\ \mathrm{W} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \mathrm{X} \begin{bmatrix} U \\ V \\ W \end{bmatrix}$$

$$\mathrm{U} = J_{11}U + J_{12}V + J_{13}W$$
$$\mathrm{V} = J_{21}U + J_{22}V + J_{23}W$$
$$\mathrm{W} = J_{31}U + J_{32}V + J_{33}W$$

However, we need to perform just the opposite coordinate transformation. The coordinate transformation matrix $\mathbf{K}_{3x3}$ is defined as inverse of transformation matrix $\mathbf{J}_{3x3}$.

$$\mathbf{K}_{3X3} = \mathbf{J}^{-1}_{3X3} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}^{-1} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix}$$

The coordinate transformation matrix $\mathbf{K}_{3x3}$ can be used to convert laser coordinate velocities to tunnel coordinate velocities.

$$\begin{bmatrix} \mathit{U} \\ \mathit{V} \\ \mathit{W} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} X \begin{bmatrix} U \\ V \\ W \end{bmatrix}$$

$$\mathit{U} = K_{11}U + K_{12}V + K_{13}W$$
$$\mathit{V} = K_{21}U + K_{22}V + K_{23}W$$
$$\mathit{W} = K_{31}U + K_{32}V + K_{33}W$$

4. Tunnel to Model Coordinate System Transformation

In some cases it is preferred to perform an additional coordinate system transformation to obtain velocities in "Model Coordinates." For example, if the model is at an angle of attack, then the model's coordinate system would be at rotation with respect to the tunnel's coordinate system. Other model attitude angles in addition to the angle of attack, such as roll and yaw, can be used determine the transformation required to convert from tunnel to model coordinates. This section describes how this tunnel to model coordinate system transformation is performed.

The angle of attack, roll, and yaw angles are defined as follows:

α_1 angle of attack

α_2 angle of roll

α_3 angle of yaw

Velocities that are calculated in tunnel coordinates U,V,W can be transformed to model coordinate velocities *(U,V,W)*. Each of the tunnel coordinate velocities U,V,W are parallel to the tunnel's X,Y,Z axes. Each of them can be transformed to model coordinates where each of the model coordinate velocities *U,V,W* would be parallel to the model's X,Y,Z axes. If the model were at angle of attack ($\alpha_1 \neq 0$), then the tunnel coordinate velocities can be defined in terms of the model coordinate velocities using the follow equations:

$$
\begin{aligned}
\mathrm{U} &= + \cos(\alpha_1)\,U + 0\,V + \sin(\alpha_1)\,W \\
\mathrm{V} &= + 0\,U + 1\,V + 0\,W \\
\mathrm{W} &= - \sin(\alpha_1)\,U + 0\,V + \cos(\alpha_1)\,W
\end{aligned}
$$

If the model were at angle of roll ($\alpha_2 \neq 0$), then the tunnel coordinate velocities can be defined in terms of the model coordinate velocities using the follow equations:

$$
\begin{aligned}
\mathrm{U} &= + 1\,U + 0\,V + 0\,W \\
\mathrm{V} &= + 0\,U + \cos(\alpha_2)\,V - \sin(\alpha_2)\,W \\
\mathrm{W} &= + 0\,U + \sin(\alpha_2)\,V + \cos(\alpha_2)\,W
\end{aligned}
$$

If the model were at angle of yaw ($\alpha_3 \neq 0$), then the tunnel coordinate velocities can be defined

in terms of the model coordinate velocities using the follow equations:

$$
\begin{aligned}
\mathrm{U} &= +\cos(\alpha_3)\,U - \sin(\alpha_3)\,V + W \\
\mathrm{V} &= +\sin(\alpha_3)\,U + \cos(\alpha_3)\,V + 0\,W \\
\mathrm{W} &= +U + 0\,V + W
\end{aligned}
$$

The model to tunnel coordinate system transformation matrices for each of the three sets of equations are defined as follows:

$$
\mathbf{J}_{\alpha 1} = \begin{bmatrix} +\cos(\alpha_1) & 0 & +\sin(\alpha_1) \\ 0 & 1 & 0 \\ -\sin(\alpha_1) & 0 & +\cos(\alpha_1) \end{bmatrix}
$$

$$
\mathbf{J}_{\alpha 2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & +\cos(\alpha_2) & -\sin(\alpha_2) \\ 0 & +\sin(\alpha_2) & +\cos(\alpha_2) \end{bmatrix}
$$

$$
\mathbf{J}_{\alpha 3} = \begin{bmatrix} +\cos(\alpha_3) & -\sin(\alpha_3) & 0 \\ +\sin(\alpha_3) & +\cos(\alpha_3) & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

If the angles of attack, roll, and yaw are used in combination, then an equivalent model to tunnel coordinate system transformation matrix can be obtained by computing the cross products of the three individual transformations.

$$
\mathbf{J}_{3\mathrm{X}3} = \mathbf{J}_{\alpha 3} \times \mathbf{J}_{\alpha 2} \times \mathbf{J}_{\alpha 1}
$$

$$
\mathbf{J}_{3\mathrm{X}3} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}
$$

$$
\mathbf{J}_{3\mathrm{X}3} = \begin{bmatrix} +\cos(\alpha_3) & -\sin(\alpha_3) & 0 \\ +\sin(\alpha_3) & +\cos(\alpha_3) & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & +\cos(\alpha_2) & -\sin(\alpha_2) \\ 0 & +\sin(\alpha_2) & +\cos(\alpha_2) \end{bmatrix} \times \begin{bmatrix} +\cos(\alpha_1) & 0 & +\sin(\alpha_1) \\ 0 & 1 & 0 \\ -\sin(\alpha_1) & 0 & +\cos(\alpha_1) \end{bmatrix}
$$

The coordinate transformation matrix $\mathbf{J}_{3x3}$ can be used to convert model coordinate velocities to tunnel coordinate velocities.

$$\begin{bmatrix} \mathrm{U} \\ \mathrm{V} \\ \mathrm{W} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \mathrm{X} \begin{bmatrix} U \\ V \\ W \end{bmatrix}$$

$$\begin{aligned} \mathrm{U} &= J_{11}U + J_{12}V + J_{13}W \\ \mathrm{V} &= J_{21}U + J_{22}V + J_{23}W \\ \mathrm{W} &= J_{31}U + J_{32}V + J_{33}W \end{aligned}$$

However, we need to perform just the opposite coordinate transformation. The coordinate transformation matrix $\mathbf{K}_{3x3}$ is defined as inverse of transformation matrix $\mathbf{J}_{3x3}$.

$$\mathbf{K}_{3X3} = \mathbf{J}^{-1}_{3X3} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}^{-1} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix}$$

The coordinate transformation matrix $\mathbf{K}_{3x3}$ can be used to convert tunnel coordinate velocities to model coordinate velocities.

$$\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \mathrm{X} \begin{bmatrix} \mathrm{U} \\ \mathrm{V} \\ \mathrm{W} \end{bmatrix}$$

$$\begin{aligned} U &= K_{11}\mathrm{U} + K_{12}\mathrm{V} + K_{13}\mathrm{W} \\ V &= K_{21}\mathrm{U} + K_{22}\mathrm{V} + K_{23}\mathrm{W} \\ W &= K_{31}\mathrm{U} + K_{32}\mathrm{V} + K_{33}\mathrm{W}\ . \end{aligned}$$

5. Data Reduction Equations

This section contains the equations that are used to calculate the average, standard deviation as well as normal and shear stress terms for the velocity and voltage data. Equations are included for both the original and transformed to coordinate systems. Normal text is used in the equations for variables that represent the original coordinate system while italicized text is used for variables that represent the transformed to coordinate system.

The following equations are used to calculate the velocity and voltage averages:

$$\overline{\mathrm{U}} = \frac{\sum_{i=1}^{N} \mathrm{U}_i}{N} \qquad \overline{\mathrm{V}} = \frac{\sum_{i=1}^{N} \mathrm{V}_i}{N} \qquad \overline{\mathrm{W}} = \frac{\sum_{i=1}^{N} \mathrm{W}_i}{N} \qquad \overline{\mathrm{A}} = \frac{\sum_{i=1}^{N} \mathrm{A}_i}{N} \qquad \overline{\mathrm{B}} = \frac{\sum_{i=1}^{N} \mathrm{B}_i}{N}$$

$$\overline{U} = \frac{\sum_{i=1}^{N} U_i}{N} \qquad \overline{V} = \frac{\sum_{i=1}^{N} V_i}{N} \qquad \overline{W} = \frac{\sum_{i=1}^{N} W_i}{N} \qquad \overline{A} = \frac{\sum_{i=1}^{N} A_i}{N} \qquad \overline{B} = \frac{\sum_{i=1}^{N} B_i}{N}$$

The following equations are used to calculate their standard deviations:

$$\mathrm{U}' = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{U}_i - \overline{\mathrm{U}})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} \mathrm{U}_i^2}{N} - \overline{\mathrm{U}}^2} \qquad U' = \sqrt{\frac{\sum_{i=1}^{N} (U_i - \overline{U})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} U_i^2}{N} - \overline{U}^2}$$

$$\mathrm{V}' = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{V}_i - \overline{\mathrm{V}})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} \mathrm{V}_i^2}{N} - \overline{\mathrm{V}}^2} \qquad V' = \sqrt{\frac{\sum_{i=1}^{N} (V_i - \overline{V})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} V_i^2}{N} - \overline{V}^2}$$

$$\mathrm{W}' = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{W}_i - \overline{\mathrm{W}})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} \mathrm{W}_i^2}{N} - \overline{\mathrm{W}}^2} \qquad W' = \sqrt{\frac{\sum_{i=1}^{N} (W_i - \overline{W})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} W_i^2}{N} - \overline{W}^2}$$

$$\mathrm{A}' = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{A}_i - \overline{\mathrm{A}})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} \mathrm{A}_i^2}{N} - \overline{\mathrm{A}}^2} \qquad A' = \sqrt{\frac{\sum_{i=1}^{N} (A_i - \overline{A})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} A_i^2}{N} - \overline{A}^2}$$

$$\mathrm{B}' = \sqrt{\frac{\sum_{i=1}^{N} (\mathrm{B}_i - \overline{\mathrm{B}})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} \mathrm{B}_i^2}{N} - \overline{\mathrm{B}}^2} \qquad B' = \sqrt{\frac{\sum_{i=1}^{N} (B_i - \overline{B})^2}{N}} = \sqrt{\frac{\sum_{i=1}^{N} B_i^2}{N} - \overline{B}^2}$$

The following equations are used to calculate the normal and shear stress terms for all of the relevant velocity:velocity, velocity:voltage, and voltage:voltage combinations:

$$\overline{U'U'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N}U_iU_i}{N} - \overline{U}\,\overline{U} = \frac{\sum_{i=1}^{N}U_i^2}{N} - \overline{U}^2$$

$$\overline{U'V'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N}U_iV_i}{N} - \overline{U}\,\overline{V}$$

$$\overline{U'W'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N}U_iW_i}{N} - \overline{U}\,\overline{W}$$

$$\overline{V'U'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N}V_iU_i}{N} - \overline{V}\,\overline{U}$$

$$\overline{V'V'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N}V_iV_i}{N} - \overline{V}\,\overline{V} = \frac{\sum_{i=1}^{N}V_i^2}{N} - \overline{V}^2$$

$$\overline{V'W'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N}V_iW_i}{N} - \overline{V}\,\overline{W}$$

$$\overline{W'U'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N}W_iU_i}{N} - \overline{W}\,\overline{U}$$

$$\overline{W'V'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N}W_iV_i}{N} - \overline{W}\,\overline{V}$$

$$\overline{W'W'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N}W_iW_i}{N} - \overline{W}\,\overline{W} = \frac{\sum_{i=1}^{N}W_i^2}{N} - \overline{W}^2$$

$$\overline{U'U'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N} U_i U_i}{N} - \overline{U}\,\overline{U} = \frac{\sum_{i=1}^{N} U_i^2}{N} - \overline{U}^2$$

$$\overline{U'V'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N} U_i V_i}{N} - \overline{U}\,\overline{V}$$

$$\overline{U'W'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N} U_i W_i}{N} - \overline{U}\,\overline{W}$$

$$\overline{V'U'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N} V_i U_i}{N} - \overline{V}\,\overline{U}$$

$$\overline{V'V'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N} V_i V_i}{N} - \overline{V}\,\overline{V} = \frac{\sum_{i=1}^{N} V_i^2}{N} - \overline{V}^2$$

$$\overline{V'W'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N} V_i W_i}{N} - \overline{V}\,\overline{W}$$

$$\overline{W'U'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N} W_i U_i}{N} - \overline{W}\,\overline{U}$$

$$\overline{W'V'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N} W_i V_i}{N} - \overline{W}\,\overline{V}$$

$$\overline{W'W'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(W_i-\overline{W})}{N} = \frac{\sum_{i=1}^{N} W_i W_i}{N} - \overline{W}\,\overline{W} = \frac{\sum_{i=1}^{N} W_i^2}{N} - \overline{W}^2$$

$$\overline{A'A'} = \frac{\sum_{i=1}^{N}(A_i-\overline{A})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} A_i A_i}{N} - \overline{A}\,\overline{A} = \frac{\sum_{i=1}^{N} A_i^2}{N} - \overline{A}^2$$

$$\overline{B'B'} = \frac{\sum_{i=1}^{N}(B_i-\overline{B})(B_i-\overline{B})}{N} = \frac{\sum_{i=1}^{N} B_i B_i}{N} - \overline{B}\,\overline{B} = \frac{\sum_{i=1}^{N} B_i^2}{N} - \overline{B}^2$$

$$\overline{A'B'} = \frac{\sum_{i=1}^{N}(A_i-\overline{A})(B_i-\overline{B})}{N} = \frac{\sum_{i=1}^{N} A_i B_i}{N} - \overline{A}\,\overline{B}$$

$$\overline{U'A'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} U_i A_i}{N} - \overline{U}\,\overline{A}$$

$$\overline{V'A'} = \frac{\sum_{i=1}^{N}(V_i-\overline{V})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} V_i A_i}{N} - \overline{V}\,\overline{A}$$

$$\overline{W'A'} = \frac{\sum_{i=1}^{N}(W_i-\overline{W})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} W_i A_i}{N} - \overline{W}\,\overline{A}$$

$$\overline{A'A'} = \frac{\sum_{i=1}^{N} (A_i-\overline{A})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} A_i A_i}{N} - \overline{A}\,\overline{A} = \frac{\sum_{i=1}^{N} A_i^2}{N} - \overline{A}^2$$

$$\overline{B'B'} = \frac{\sum_{i=1}^{N} (B_i-\overline{B})(B_i-\overline{B})}{N} = \frac{\sum_{i=1}^{N} B_i B_i}{N} - \overline{B}\,\overline{B} = \frac{\sum_{i=1}^{N} B_i^2}{N} - \overline{B}^2$$

$$\overline{A'B'} = \frac{\sum_{i=1}^{N} (A_i-\overline{A})(B_i-\overline{B})}{N} = \frac{\sum_{i=1}^{N} A_i B_i}{N} - \overline{A}\,\overline{B}$$

$$\overline{U'A'} = \frac{\sum_{i=1}^{N} (U_i-\overline{U})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} U_i A_i}{N} - \overline{U}\,\overline{A}$$

$$\overline{V'A'} = \frac{\sum_{i=1}^{N} (V_i-\overline{V})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} V_i A_i}{N} - \overline{V}\,\overline{A}$$

$$\overline{W'A'} = \frac{\sum_{i=1}^{N} (W_i-\overline{W})(A_i-\overline{A})}{N} = \frac{\sum_{i=1}^{N} W_i A_i}{N} - \overline{W}\,\overline{A}$$

6. Coordinate System Transformation Equations

The following equations are used to convert the instantaneous as well as the average velocities from the original to the transformed to coordinate system. The instantaneous and average values for the voltage data are the same in either coordinate systems:

$$U_i = K_{11}U_i + K_{12}V_i + K_{13}W_i \qquad \overline{U} = K_{11}\overline{U} + K_{12}\overline{V} + K_{13}\overline{W}$$

$$V_i = K_{21}U_i + K_{22}V_i + K_{23}W_i \qquad \overline{V} = K_{21}\overline{U} + K_{22}\overline{V} + K_{23}\overline{W}$$

$$W_i = K_{31}U_i + K_{32}V_i + K_{33}W_i \qquad \overline{W} = K_{31}\overline{U} + K_{32}\overline{V} + K_{33}\overline{W}$$

$$A_i = A_i \qquad \overline{A} = \overline{A}$$

$$B_i = B_i \qquad \overline{B} = \overline{B}$$

The following equations are used to convert the velocity:voltage and voltage:voltage normal and shear stress terms from the original to the transformed to coordinate system. The voltage:voltage normal and shear stress terms are the same in either coordinate systems:

$$\overline{U'A'} = K_{11}\overline{U'A'} + K_{12}\overline{V'A'} + K_{13}\overline{W'A'}$$

$$\overline{V'A'} = K_{21}\overline{U'A'} + K_{22}\overline{V'A'} + K_{23}\overline{W'A'}$$

$$\overline{W'A'} = K_{31}\overline{U'A'} + K_{32}\overline{V'A'} + K_{33}\overline{W'A'}$$

$$\overline{A'A'} = \overline{A'A'}$$

$$\overline{B'B'} = \overline{B'B'}$$

$$\overline{A'B'} = \overline{A'B'}$$

The following equations are used to convert the velocity:velocity normal and shear stress terms from the original to the transformed to coordinate system:

$$
\begin{aligned}
\overline{U'U'} = {} & K_{11}K_{11}\overline{U'U'} + K_{11}K_{12}\overline{U'V'} + K_{11}K_{13}\overline{U'W'} \\
& + K_{12}K_{11}\overline{V'U'} + K_{12}K_{12}\overline{V'V'} + K_{12}K_{13}\overline{V'W'} \\
& + K_{13}K_{11}\overline{W'U'} + K_{13}K_{12}\overline{W'V'} + K_{13}K_{13}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{U'V'} = {} & K_{11}K_{21}\overline{U'U'} + K_{11}K_{22}\overline{U'V'} + K_{11}K_{23}\overline{U'W'} \\
& + K_{12}K_{21}\overline{V'U'} + K_{12}K_{22}\overline{V'V'} + K_{12}K_{23}\overline{V'W'} \\
& + K_{13}K_{21}\overline{W'U'} + K_{13}K_{22}\overline{W'V'} + K_{13}K_{23}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{U'W'} = {} & K_{11}K_{31}\overline{U'U'} + K_{11}K_{32}\overline{U'V'} + K_{11}K_{33}\overline{U'W'} \\
& + K_{12}K_{31}\overline{V'U'} + K_{12}K_{32}\overline{V'V'} + K_{12}K_{33}\overline{V'W'} \\
& + K_{13}K_{31}\overline{W'U'} + K_{13}K_{32}\overline{W'V'} + K_{13}K_{33}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{V'U'} = {} & K_{21}K_{11}\overline{U'U'} + K_{21}K_{12}\overline{U'V'} + K_{21}K_{13}\overline{U'W'} \\
& + K_{22}K_{11}\overline{V'U'} + K_{22}K_{12}\overline{V'V'} + K_{22}K_{13}\overline{V'W'} \\
& + K_{23}K_{11}\overline{W'U'} + K_{23}K_{12}\overline{W'V'} + K_{23}K_{13}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{V'V'} = {} & K_{21}K_{21}\overline{U'U'} + K_{21}K_{22}\overline{U'V'} + K_{21}K_{23}\overline{U'W'} \\
& + K_{22}K_{21}\overline{V'U'} + K_{22}K_{22}\overline{V'V'} + K_{22}K_{23}\overline{V'W'} \\
& + K_{23}K_{21}\overline{W'U'} + K_{23}K_{22}\overline{W'V'} + K_{23}K_{23}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{V'W'} = {} & K_{21}K_{31}\overline{U'U'} + K_{21}K_{32}\overline{U'V'} + K_{21}K_{33}\overline{U'W'} \\
& + K_{22}K_{31}\overline{V'U'} + K_{22}K_{32}\overline{V'V'} + K_{22}K_{33}\overline{V'W'} \\
& + K_{23}K_{31}\overline{W'U'} + K_{23}K_{32}\overline{W'V'} + K_{23}K_{33}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{W'U'} = {} & K_{31}K_{11}\overline{U'U'} + K_{31}K_{12}\overline{U'V'} + K_{31}K_{13}\overline{U'W'} \\
& + K_{32}K_{11}\overline{V'U'} + K_{32}K_{12}\overline{V'V'} + K_{32}K_{13}\overline{V'W'} \\
& + K_{33}K_{11}\overline{W'U'} + K_{33}K_{12}\overline{W'V'} + K_{33}K_{13}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{W'V'} = {} & K_{31}K_{21}\overline{U'U'} + K_{31}K_{22}\overline{U'V'} + K_{31}K_{23}\overline{U'W'} \\
& + K_{32}K_{21}\overline{V'U'} + K_{32}K_{22}\overline{V'V'} + K_{32}K_{23}\overline{V'W'} \\
& + K_{33}K_{21}\overline{W'U'} + K_{33}K_{22}\overline{W'V'} + K_{33}K_{23}\overline{W'W'}
\end{aligned}
$$

$$
\begin{aligned}
\overline{W'W'} = {} & K_{31}K_{31}\overline{U'U'} + K_{31}K_{32}\overline{U'V'} + K_{31}K_{33}\overline{U'W'} \\
& + K_{32}K_{31}\overline{V'U'} + K_{32}K_{32}\overline{V'V'} + K_{32}K_{33}\overline{V'W'} \\
& + K_{33}K_{31}\overline{W'U'} + K_{33}K_{32}\overline{W'V'} + K_{33}K_{33}\overline{W'W'}
\end{aligned}
$$

7. Proofs for Coordinate System Transformation Equations

This section contains proofs demonstrating that we can perform the coordinate system transformations on the reduced averaged data without having to perform the transformation on the instantaneous values. This saves costly run time because there are typically thousands of instantaneous values that contribute one averaged value.

The following equations show how the average velocities from the original coordinate system can be used along with the coordinate transformation matrix to provide velocities in the new transformed to coordinate system:

$$\overline{U} = \frac{\sum_{i=1}^{N} U_i}{N}$$

$$U_i = K_{11}\mathrm{U}_i + K_{12}\mathrm{V}_i + K_{13}\mathrm{W}_i$$

$$\overline{U} = \frac{\sum_{i=1}^{N} (K_{11}\mathrm{U}_i + K_{12}\mathrm{V}_i + K_{13}\mathrm{W}_i)}{N}$$

$$\overline{U} = \frac{\sum_{i=1}^{N} K_{11}\mathrm{U}_i}{N} + \frac{\sum_{i=1}^{N} K_{12}\mathrm{V}_i}{N} + \frac{\sum_{i=1}^{N} K_{13}\mathrm{W}_i}{N}$$

$$\overline{U} = K_{11}\frac{\sum_{i=1}^{N} \mathrm{U}_i}{N} + K_{12}\frac{\sum_{i=1}^{N} \mathrm{V}_i}{N} + K_{13}\frac{\sum_{i=1}^{N} \mathrm{W}_i}{N}$$

$$\overline{U} = K_{11}\overline{\mathrm{U}} + K_{12}\overline{\mathrm{V}} + K_{13}\overline{\mathrm{W}}$$

With similar proofs we can show that the following equations apply:

$$\overline{V} = K_{21}\overline{\mathrm{U}} + K_{22}\overline{\mathrm{V}} + K_{23}\overline{\mathrm{W}}$$

$$\overline{W} = K_{31}\overline{\mathrm{U}} + K_{32}\overline{\mathrm{V}} + K_{33}\overline{\mathrm{W}}$$

The following equations show how the velocity:velocity normal and shear stress terms from the original coordinate system can be used along with the coordinate transformation matrix to provide velocity:velocity normal stress terms in the new transformed to coordinate system:

$$\overline{U'U'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(U_i-\overline{U})}{N} = \frac{\sum_{i=1}^{N} U_i U_i}{N} - \overline{U}\,\overline{U}$$

$$U_i = \mathrm{K}_{11}\mathrm{U}_i + \mathrm{K}_{12}\mathrm{V}_i + \mathrm{K}_{13}\mathrm{W}_i \qquad \overline{U} = \mathrm{K}_{11}\overline{\mathrm{U}} + \mathrm{K}_{12}\overline{\mathrm{V}} + \mathrm{K}_{13}\overline{\mathrm{W}}$$

$$\overline{U'U'} = \frac{\sum_{i=1}^{N}(\mathrm{K}_{11}\mathrm{U}_i+\mathrm{K}_{12}\mathrm{V}_i+\mathrm{K}_{13}\mathrm{W}_i)(\mathrm{K}_{11}\mathrm{U}_i+\mathrm{K}_{12}\mathrm{V}_i+\mathrm{K}_{13}\mathrm{W}_i)}{N} - (\mathrm{K}_{11}\overline{\mathrm{U}}+\mathrm{K}_{12}\overline{\mathrm{V}}+\mathrm{K}_{13}\overline{\mathrm{W}})(\mathrm{K}_{11}\overline{\mathrm{U}}+\mathrm{K}_{12}\overline{\mathrm{V}}+\mathrm{K}_{13}\overline{\mathrm{W}})$$

$$\overline{U'U'} = \frac{\sum_{i=1}^{N}\begin{pmatrix} \mathrm{K}_{11}\mathrm{K}_{11}\mathrm{U}_i\mathrm{U}_i+\mathrm{K}_{11}\mathrm{K}_{12}\mathrm{U}_i\mathrm{V}_i+\mathrm{K}_{11}\mathrm{K}_{13}\mathrm{U}_i\mathrm{W}_i \\ +\mathrm{K}_{12}\mathrm{K}_{11}\mathrm{V}_i\mathrm{U}_i+\mathrm{K}_{12}\mathrm{K}_{12}\mathrm{V}_i\mathrm{V}_i+\mathrm{K}_{12}\mathrm{K}_{13}\mathrm{V}_i\mathrm{W}_i \\ +\mathrm{K}_{13}\mathrm{K}_{11}\mathrm{W}_i\mathrm{U}_i+\mathrm{K}_{13}\mathrm{K}_{12}\mathrm{W}_i\mathrm{V}_i+\mathrm{K}_{13}\mathrm{K}_{13}\mathrm{W}_i\mathrm{W}_i \end{pmatrix}}{N} - \begin{pmatrix} \mathrm{K}_{11}\mathrm{K}_{11}\overline{\mathrm{U}}\,\overline{\mathrm{U}}+\mathrm{K}_{11}\mathrm{K}_{12}\overline{\mathrm{U}}\,\overline{\mathrm{V}}+\mathrm{K}_{11}\mathrm{K}_{13}\overline{\mathrm{U}}\,\overline{\mathrm{W}} \\ +\mathrm{K}_{12}\mathrm{K}_{11}\overline{\mathrm{V}}\,\overline{\mathrm{U}}+\mathrm{K}_{12}\mathrm{K}_{12}\overline{\mathrm{V}}\,\overline{\mathrm{V}}+\mathrm{K}_{12}\mathrm{K}_{13}\overline{\mathrm{V}}\,\overline{\mathrm{W}} \\ +\mathrm{K}_{13}\mathrm{K}_{11}\overline{\mathrm{W}}\,\overline{\mathrm{U}}+\mathrm{K}_{13}\mathrm{K}_{12}\overline{\mathrm{W}}\,\overline{\mathrm{V}}+\mathrm{K}_{13}\mathrm{K}_{13}\overline{\mathrm{W}}\,\overline{\mathrm{W}} \end{pmatrix}$$

$$\begin{aligned}
\overline{U'U'} = {} & \mathrm{K}_{11}\mathrm{K}_{11}\left(\frac{\sum_{i=1}^{N}\mathrm{U}_i\mathrm{U}_i}{N} - \overline{\mathrm{U}}\,\overline{\mathrm{U}}\right) + \mathrm{K}_{11}\mathrm{K}_{12}\left(\frac{\sum_{i=1}^{N}\mathrm{U}_i\mathrm{V}_i}{N} - \overline{\mathrm{U}}\,\overline{\mathrm{V}}\right) + \mathrm{K}_{11}\mathrm{K}_{13}\left(\frac{\sum_{i=1}^{N}\mathrm{U}_i\mathrm{W}_i}{N} - \overline{\mathrm{U}}\,\overline{\mathrm{W}}\right) \\
& + \mathrm{K}_{12}\mathrm{K}_{11}\left(\frac{\sum_{i=1}^{N}\mathrm{V}_i\mathrm{U}_i}{N} - \overline{\mathrm{V}}\,\overline{\mathrm{U}}\right) + \mathrm{K}_{12}\mathrm{K}_{12}\left(\frac{\sum_{i=1}^{N}\mathrm{V}_i\mathrm{V}_i}{N} - \overline{\mathrm{V}}\,\overline{\mathrm{V}}\right) + \mathrm{K}_{12}\mathrm{K}_{13}\left(\frac{\sum_{i=1}^{N}\mathrm{V}_i\mathrm{W}_i}{N} - \overline{\mathrm{V}}\,\overline{\mathrm{W}}\right) \\
& + \mathrm{K}_{13}\mathrm{K}_{11}\left(\frac{\sum_{i=1}^{N}\mathrm{W}_i\mathrm{U}_i}{N} - \overline{\mathrm{W}}\,\overline{\mathrm{U}}\right) + \mathrm{K}_{13}\mathrm{K}_{12}\left(\frac{\sum_{i=1}^{N}\mathrm{W}_i\mathrm{V}_i}{N} - \overline{\mathrm{W}}\,\overline{\mathrm{V}}\right) + \mathrm{K}_{13}\mathrm{K}_{13}\left(\frac{\sum_{i=1}^{N}\mathrm{W}_i\mathrm{W}_i}{N} - \overline{\mathrm{W}}\,\overline{\mathrm{W}}\right)
\end{aligned}$$

$$\begin{aligned}
\overline{U'U'} = {} & \mathrm{K}_{11}\mathrm{K}_{11}\overline{\mathrm{U'U'}} + \mathrm{K}_{11}\mathrm{K}_{12}\overline{\mathrm{U'V'}} + \mathrm{K}_{11}\mathrm{K}_{13}\overline{\mathrm{U'W'}} \\
& + \mathrm{K}_{12}\mathrm{K}_{11}\overline{\mathrm{V'U'}} + \mathrm{K}_{12}\mathrm{K}_{12}\overline{\mathrm{V'V'}} + \mathrm{K}_{12}\mathrm{K}_{13}\overline{\mathrm{V'W'}} \\
& + \mathrm{K}_{13}\mathrm{K}_{11}\overline{\mathrm{W'U'}} + \mathrm{K}_{13}\mathrm{K}_{12}\overline{\mathrm{W'V'}} + \mathrm{K}_{13}\mathrm{K}_{13}\overline{\mathrm{W'W'}}
\end{aligned}$$

The following equations show how the velocity:velocity normal and shear stress terms from the original coordinate system can be used along with the coordinate transformation matrix to provide velocity:velocity shear stress terms in the new transformed to coordinate system:

$$\overline{U'V'} = \frac{\sum_{i=1}^{N}(U_i-\overline{U})(V_i-\overline{V})}{N} = \frac{\sum_{i=1}^{N} U_i V_i}{N} - \overline{U}\,\overline{V}$$

$$U_i = K_{11}U_i + K_{12}V_i + K_{13}W_i \qquad \overline{U} = K_{11}\overline{U} + K_{12}\overline{V} + K_{13}\overline{W}$$

$$V_i = K_{21}U_i + K_{22}V_i + K_{23}W_i \qquad \overline{V} = K_{21}\overline{U} + K_{22}\overline{V} + K_{23}\overline{W}$$

$$\overline{U'V'} = \frac{\sum_{i=1}^{N}(K_{11}U_i+K_{12}V_i+K_{13}W_i)(K_{21}U_i+K_{22}V_i+K_{23}W_i)}{N} - (K_{11}\overline{U}+K_{12}\overline{V}+K_{13}\overline{W})(K_{21}\overline{U}+K_{22}\overline{V}+K_{23}\overline{W})$$

$$\overline{U'V'} = \frac{\sum_{i=1}^{N}\begin{pmatrix} K_{11}K_{21}U_iU_i+K_{11}K_{22}U_iV_i+K_{11}K_{23}U_iW_i \\ +K_{12}K_{21}V_iU_i+K_{12}K_{22}V_iV_i+K_{12}K_{23}V_iW_i \\ +K_{13}K_{21}W_iU_i+K_{13}K_{22}W_iV_i+K_{13}K_{23}W_iW_i \end{pmatrix}}{N} - \begin{pmatrix} K_{11}K_{21}\overline{U}\,\overline{U}+K_{11}K_{22}\overline{U}\,\overline{V}+K_{11}K_{23}\overline{U}\,\overline{W} \\ +K_{12}K_{21}\overline{V}\,\overline{U}+K_{12}K_{22}\overline{V}\,\overline{V}+K_{12}K_{23}\overline{V}\,\overline{W} \\ +K_{13}K_{21}\overline{W}\,\overline{U}+K_{13}K_{22}\overline{W}\,\overline{V}+K_{13}K_{23}\overline{W}\,\overline{W} \end{pmatrix}$$

$$\begin{aligned} \overline{U'V'} = {} & K_{11}K_{21}\left(\frac{\sum_{i=1}^{N} U_iU_i}{N} - \overline{U}\,\overline{U}\right) + K_{11}K_{22}\left(\frac{\sum_{i=1}^{N} U_iV_i}{N} - \overline{U}\,\overline{V}\right) + K_{11}K_{23}\left(\frac{\sum_{i=1}^{N} U_iW_i}{N} - \overline{U}\,\overline{W}\right) \\ & + K_{12}K_{21}\left(\frac{\sum_{i=1}^{N} V_iU_i}{N} - \overline{V}\,\overline{U}\right) + K_{12}K_{22}\left(\frac{\sum_{i=1}^{N} V_iV_i}{N} - \overline{V}\,\overline{V}\right) + K_{12}K_{23}\left(\frac{\sum_{i=1}^{N} V_iW_i}{N} - \overline{V}\,\overline{W}\right) \\ & + K_{13}K_{21}\left(\frac{\sum_{i=1}^{N} W_iU_i}{N} - \overline{W}\,\overline{U}\right). + K_{13}K_{22}\left(\frac{\sum_{i=1}^{N} W_iV_i}{N} - \overline{W}\,\overline{V}\right) + K_{13}K_{23}\left(\frac{\sum_{i=1}^{N} W_iW_i}{N} - \overline{W}\,\overline{W}\right) \end{aligned}$$

$$\begin{aligned} \overline{U'V'} = {} & K_{11}K_{21}\overline{U'U'} + K_{11}K_{22}\overline{U'V'} + K_{11}K_{23}\overline{U'W'} \\ & + K_{12}K_{21}\overline{V'U'} + K_{12}K_{22}\overline{V'V'} + K_{12}K_{23}\overline{V'W'} \\ & + K_{13}K_{21}\overline{W'U'} + K_{13}K_{22}\overline{W'V'} + K_{13}K_{23}\overline{W'W'} \end{aligned}$$

The following equations show how the velocity:voltage shear stress terms from the original coordinate system can be used along with the coordinate transformation matrix to provide velocity:voltage shear stress terms in the new transformed to coordinate system:

$$\overline{\mathcal{U}'\mathcal{A}'} = \frac{\sum_{i=1}^{N}(\mathcal{U}_i-\overline{\mathcal{U}})(\mathcal{A}_i-\overline{\mathcal{A}})}{N} = \frac{\sum_{i=1}^{N}\mathcal{U}_i\mathcal{A}_i}{N} - \overline{\mathcal{U}}\,\overline{\mathcal{A}}$$

$$\mathcal{U}_i = K_{11}U_i + K_{12}V_i + K_{13}W_i \qquad \overline{\mathcal{U}} = K_{11}\overline{U} + K_{12}\overline{V} + K_{13}\overline{W}$$

$$\mathcal{A}_i = A_i \qquad \overline{\mathcal{A}} = \overline{A}$$

$$\overline{\mathcal{U}'\mathcal{A}'} = \frac{\sum_{i=1}^{N}(K_{11}U_i+K_{12}V_i+K_{13}W_i)(A_i)}{N} - (K_{11}\overline{U}+K_{12}\overline{V}+K_{13}\overline{W})(\overline{A})$$

$$\overline{\mathcal{U}'\mathcal{A}'} = \frac{\sum_{i=1}^{N}(K_{11}U_iA_i+K_{12}V_iA_i+K_{13}W_iA_i)}{N} - (K_{11}\overline{U}\,\overline{A}+K_{12}\overline{V}\,\overline{A}+K_{13}\overline{W}\,\overline{A})$$

$$\overline{\mathcal{U}'\mathcal{A}'} = K_{11}\left(\frac{\sum_{i=1}^{N}U_iA_i}{N} - \overline{U}\,\overline{A}\right) + K_{12}\left(\frac{\sum_{i=1}^{N}V_iA_i}{N} - \overline{V}\,\overline{A}\right) + K_{13}\left(\frac{\sum_{i=1}^{N}W_iA_i}{N} - \overline{W}\,\overline{A}\right)$$

$$\overline{\mathcal{U}'\mathcal{A}'} = K_{11}\overline{U'A'} + K_{12}\overline{V'A'} + K_{13}\overline{W'A'}$$

8. Matrix Notation for Coordinate System Transformation Equations

This section shows how the equations of section 1.6 can be represented in matrix notation. The matrix notation for the coordinate system transformation is an elegant way to show the multitude of complex equations in compact and concise format. The rest of this page contains various matrix definitions:

$$\mathbf{J}_{3X3} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \qquad \mathbf{K}_{3X3} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} \overline{U} \\ \overline{V} \\ \overline{W} \end{bmatrix} \qquad \mathbf{S}_i = \begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix} \qquad \mathbf{F} = \begin{bmatrix} \overline{U'A'} \\ \overline{V'A'} \\ \overline{W'A'} \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} \overline{U'B'} \\ \overline{V'B'} \\ \overline{W'B'} \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} \overline{U} \\ \overline{V} \\ \overline{W} \end{bmatrix} \qquad \mathbf{R}_i = \begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} \overline{U'A'} \\ \overline{V'A'} \\ \overline{W'A'} \end{bmatrix} \qquad \mathbf{I} = \begin{bmatrix} \overline{U'B'} \\ \overline{V'B'} \\ \overline{W'B'} \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} \overline{U'U'} \\ \overline{U'V'} \\ \overline{U'W'} \\ \overline{V'U'} \\ \overline{V'V'} \\ \overline{V'W'} \\ \overline{W'U'} \\ \overline{W'V'} \\ \overline{W'W'} \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} \overline{U'U'} \\ \overline{U'V'} \\ \overline{U'W'} \\ \overline{V'U'} \\ \overline{V'V'} \\ \overline{V'W'} \\ \overline{W'U'} \\ \overline{W'V'} \\ \overline{W'W'} \end{bmatrix}$$

$$\mathbf{K}_{9X9} = \begin{bmatrix} K_{11}K_{11} & K_{11}K_{12} & K_{11}K_{13} & K_{12}K_{11} & K_{12}K_{12} & K_{12}K_{13} & K_{13}K_{11} & K_{13}K_{12} & K_{13}K_{13} \\ K_{11}K_{21} & K_{11}K_{22} & K_{11}K_{23} & K_{12}K_{21} & K_{12}K_{22} & K_{12}K_{23} & K_{13}K_{21} & K_{13}K_{22} & K_{13}K_{23} \\ K_{11}K_{31} & K_{11}K_{32} & K_{11}K_{33} & K_{12}K_{31} & K_{12}K_{32} & K_{12}K_{33} & K_{13}K_{31} & K_{13}K_{32} & K_{13}K_{33} \\ K_{21}K_{11} & K_{21}K_{12} & K_{21}K_{13} & K_{22}K_{11} & K_{22}K_{12} & K_{22}K_{13} & K_{23}K_{11} & K_{23}K_{12} & K_{23}K_{13} \\ K_{21}K_{21} & K_{21}K_{22} & K_{21}K_{23} & K_{22}K_{21} & K_{22}K_{22} & K_{22}K_{23} & K_{23}K_{21} & K_{23}K_{22} & K_{23}K_{23} \\ K_{21}K_{31} & K_{21}K_{32} & K_{21}K_{33} & K_{22}K_{31} & K_{22}K_{32} & K_{22}K_{33} & K_{23}K_{31} & K_{23}K_{32} & K_{23}K_{33} \\ K_{31}K_{11} & K_{31}K_{12} & K_{31}K_{13} & K_{32}K_{11} & K_{32}K_{12} & K_{32}K_{13} & K_{33}K_{11} & K_{33}K_{12} & K_{33}K_{13} \\ K_{31}K_{21} & K_{31}K_{22} & K_{31}K_{23} & K_{32}K_{21} & K_{32}K_{22} & K_{32}K_{23} & K_{33}K_{21} & K_{33}K_{22} & K_{33}K_{23} \\ K_{31}K_{31} & K_{31}K_{32} & K_{31}K_{33} & K_{32}K_{31} & K_{32}K_{32} & K_{32}K_{33} & K_{33}K_{31} & K_{33}K_{32} & K_{33}K_{33} \end{bmatrix}$$

This page consolidates all of the coordinate transformation equations in matrix notation.

$$\mathbf{S} = \mathbf{K}_{3X3} \times \mathbf{R}$$

$$\begin{bmatrix} \overline{U} \\ \overline{V} \\ \overline{W} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \times \begin{bmatrix} \overline{U} \\ \overline{V} \\ \overline{W} \end{bmatrix}$$

$$\mathbf{S}_i = \mathbf{K}_{3X3} \times \mathbf{R}_i$$

$$\begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \times \begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix}$$

$$\mathbf{H} = \mathbf{K}_{3X3} \times \mathbf{F}$$

$$\begin{bmatrix} \overline{U'A'} \\ \overline{V'A'} \\ \overline{W'A'} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \times \begin{bmatrix} \overline{U'A'} \\ \overline{V'A'} \\ \overline{W'A'} \end{bmatrix}$$

$$\mathbf{I} = \mathbf{K}_{3X3} \times \mathbf{G}$$

$$\begin{bmatrix} \overline{U'B'} \\ \overline{V'B'} \\ \overline{W'B'} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \times \begin{bmatrix} \overline{U'B'} \\ \overline{V'B'} \\ \overline{W'B'} \end{bmatrix}$$

$$\mathbf{Q} = \mathbf{K}_{9X9} \times \mathbf{P}$$

$$\begin{bmatrix} \overline{U'U'} \\ \overline{U'V'} \\ \overline{U'W'} \\ \overline{V'U'} \\ \overline{V'V'} \\ \overline{V'W'} \\ \overline{W'U'} \\ \overline{W'V'} \\ \overline{W'W'} \end{bmatrix} = \begin{bmatrix} K_{11}K_{11} & K_{11}K_{12} & K_{11}K_{13} & K_{12}K_{11} & K_{12}K_{12} & K_{12}K_{13} & K_{13}K_{11} & K_{13}K_{12} & K_{13}K_{13} \\ K_{11}K_{21} & K_{11}K_{22} & K_{11}K_{23} & K_{12}K_{21} & K_{12}K_{22} & K_{12}K_{23} & K_{13}K_{21} & K_{13}K_{22} & K_{13}K_{23} \\ K_{11}K_{31} & K_{11}K_{32} & K_{11}K_{33} & K_{12}K_{31} & K_{12}K_{32} & K_{12}K_{33} & K_{13}K_{31} & K_{13}K_{32} & K_{13}K_{33} \\ K_{21}K_{11} & K_{21}K_{12} & K_{21}K_{13} & K_{22}K_{11} & K_{22}K_{12} & K_{22}K_{13} & K_{23}K_{11} & K_{23}K_{12} & K_{23}K_{13} \\ K_{21}K_{21} & K_{21}K_{22} & K_{21}K_{23} & K_{22}K_{21} & K_{22}K_{22} & K_{22}K_{23} & K_{23}K_{21} & K_{23}K_{22} & K_{23}K_{23} \\ K_{21}K_{31} & K_{21}K_{32} & K_{21}K_{33} & K_{22}K_{31} & K_{22}K_{32} & K_{22}K_{33} & K_{23}K_{31} & K_{23}K_{32} & K_{23}K_{33} \\ K_{31}K_{11} & K_{31}K_{12} & K_{31}K_{13} & K_{32}K_{11} & K_{32}K_{12} & K_{32}K_{13} & K_{33}K_{11} & K_{33}K_{12} & K_{33}K_{13} \\ K_{31}K_{21} & K_{31}K_{22} & K_{31}K_{23} & K_{32}K_{21} & K_{32}K_{22} & K_{32}K_{23} & K_{33}K_{21} & K_{33}K_{22} & K_{33}K_{23} \\ K_{31}K_{31} & K_{31}K_{32} & K_{31}K_{33} & K_{32}K_{31} & K_{32}K_{32} & K_{32}K_{33} & K_{33}K_{31} & K_{33}K_{32} & K_{33}K_{33} \end{bmatrix} \times \begin{bmatrix} \overline{U'U'} \\ \overline{U'V'} \\ \overline{U'W'} \\ \overline{V'U'} \\ \overline{V'V'} \\ \overline{V'W'} \\ \overline{W'U'} \\ \overline{W'V'} \\ \overline{W'W'} \end{bmatrix}$$